AD_____

Award Number: DAMD17-02-C-0119

TITLE: Generalized Linear Mixed-Effects Models in R

PRINCIPAL INVESTIGATOR: Ben C. Juricek, Ph.D.

CONTRACTING ORGANIZATION: Toyon Research Corporation
Goleta, California 93117

REPORT DATE: February 2003

TYPE OF REPORT: Final

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

DISTRIBUTION STATEMENT: Approved for Public Release;
Distribution Unlimited

The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 2003	Final (16 Aug 02 - 15 Jan 03)

4. TITLE AND SUBTITLE
Generalized Linear Mixed-Effects Models in R

5. FUNDING NUMBERS
DAMD17-02-C-0119

6. AUTHOR(S) :
Ben C. Juricek, Ph.D.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Toyon Research Corporation
Goleta, California 93117

E-Mail: bjuricek@toyon.com

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. Abstract ***(Maximum 200 Words)*** *(abstract should contain no proprietary or confidential information)*

The Nonlinear and Linear Mixed-Effects (NLME) package for the open source statistical software system R provides an effective and efficient way to analyze longitudinal data collected from nested groups of subjects when the response of interest is on a continuous scale. It does not provide methods for analyzing binary, multinomial, or ordinal responses, where a general linear mixed-effects model (GLMM) is required. We enhanced an existing R implementation for estimating a GLMM, which can estimate an approximate model using a crude numerical procedure. The R code was rewritten to take advantage of the best available numerical methods and the latest theoretical developments. Using simulated data sets, we demonstrate that the enhanced code is much faster and numerically robust. We propose an approach for modeling ordinal and multinomial data, the theory of which is less well developed than that of binomial data. The proposed approach is demonstrated using simulated and real data sets. The results illustrate the limits of the approximate procedure used in Phase I, which motivates the use of a more refined numerical method in Phase II.

14. SUBJECT TERMS:
multilevel models, categorical data, generalized linear mixed models

15. NUMBER OF PAGES
73

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

1 Introduction

A key challenge in organizational research is accounting for the existence of levels, i.e., distinguishing between the effects on individual and organizational behavior, when analyzing a particular process. Multilevel modeling has proven to be an effective method for investigating the effects at several levels within an organizational hierarchy. In today's commericial and open-source software, several tools exist that can estimate a multilevel model for a data set of continuous measurements. For data sets with binomial or categorical measurements, multilevel modeling requires estimating a generalized linear mixed model (GLMM). Currently, extant software tools for estimating a GLMM are very limited in that the numerical methods used can be slow, inaccurate, or both. In this work, Toyon Research Corporation and the University of Wisconsin are developing a software package for estimating a GLMM in R, an open-source statistical environment. For Phase I, we have developed an approach that fits an approximate model, but uses methods that can be extended for maximum likelihood (ML) methods that we will pursue in Phase II.

2 Body

As discussed in our Phase I proposal, there are three main objectives for the Phase I work:

1. Develop a preliminary version of the code for estimating a GLMM, which can be used as a foundation for Phase II.
2. Demonstrate the code using simulated data sets
3. Identify a suitable approach for modeling ordinal data sets
4. Develop a suitable interface

We elected to use the standard R interface for a function, and not to develop any specialized or graphical user interface.

As discussed in the proposal for Phase I, Penalized Quasi Likelihood (PQL) is an approximate method that can estimate a GLMM for binary data. We proposed to use this approximate method to estimate an initial solution, which can be further refined using a more accurate, but computationally intensive procedure. The Phase I effort focused on using PQL, with extensions to other methods to be developed in Phase II.

Code development was the major effort in Phase I, and consisted of some low-level R programming and theoretical work on the numerical methods for GLMM. The programming tasks consisted of re-implementing

the `lme` function in R, to take advantage of S4-classes and better linear algebra solvers. Because the `lme` function is called repeatedly in several iterations for estimating a GLMM, it is the "long pole in the tent" in terms of speed and accuracy.

The PQL method directly works for binary data, but does not clearly extend for categorical data (i.e., where the response variable can be more than two values). For Phase I, we focused on identifying a suitable method for ordinal data, which are a particularly kind of categorical data where the categories are ordered. The main intent for this work was to identify a method analogous to the PQL for binary, i.e., a method that can produce an initial estimate for an ML procedure.

Before discussing these tasks in detail, we review the pertinent technical background in the next section and describe our over-arching vision.

2.1 Technical Background

Mixed-effects models are statistical models that describe the behavior of a response variate as it relates to measured covariates. These models incorporate both *fixed-effects* parameters, which are parameters that relate to the entire population or well-defined subgroups of the population, and *random-effects*, which are random variables associated with individual experimental units. They are particularly useful with *longitudinal* data — data that are collected over time on each of several subjects. As described in the program solicitation the subjects can be grouped according to one or more nested layers of grouping. For example, soldiers can be grouped in squads that are within platoons, that are within companies, and so on.

A *multilevel* model (Goldstein, 1995; Snijders and Bosker, 1999; Raudenbush and Bryk, 2002; de Leeuw and Kreft, 2002) is another name for a mixed-effects statistical model that includes random-effects terms at one or more nested levels. Multilevel models apply to longitudinal data or to general *repeated measures* data that are not necessarily gathered over time.

Linear models are statistical models for responses that are measured on a continuous scale. They have the property that the prediction of the response is a linear function of the model parameters, or *coefficients*, and terms derived from the values of the covariates. In contrast, a *nonlinear* statistical model generally refers to models for responses measured on a continuous scale but where the prediction of the response is a nonlinear function of the model parameters.

Generalized linear models (McCullagh and Nelder, 1989) are statistical models in which a function, called the *link* function, of the expected response is modeled as a linear combination of model terms. GLMs

are frequently used to model a dichotomous response and some variations of GLMs are used for ordinal data. Typical link functions for dichotomous responses are the *logit* link or the *probit* link. There is a very efficient algorithm called iteratively reweighted least squares (IRLS) that is used to fit GLMs that do not have random-effects terms.

A linear model with both fixed-effects terms and random-effects terms is a *linear mixed-effects model.* The R function `lme` from the NLME package (Pinheiro and Bates, 2000) calculates *maximum likelihood* (ML) or *restricted maximum likelihood* (REML) parameter estimates for these models. A nonlinear model with random-effects terms is a *nonlinear mixed-effects model.* The R function `nlme`, also in the NLME package, calculates parameter estimates for these models. A generalized linear model with random-effects terms is called a *generalized linear mixed model* (GLMM) or a multilevel generalized linear model (Rodriguez, 2002). At present the NLME package does not provide capabilities for estimating the parameters in GLMMs. The R function `glmmPQL` in the MASS package provides maximum likelihood estimates of the parameters in a GLMM using an approximation method called penalized quasi-likelihood. Another function, `glmm` in the GLMMGibbs package, approximates ML estimates through a Gibbs sampler.

2.2 Technical approach

For this STTR project, we will develop and implement in R methods to determine the maximum likelihood estimates of parameters in generalized linear mixed models (GLMMs) for longitudinal data. A GLMM with one level of random effects is similar to the linear mixed-effects (LME) model

$$\begin{gathered} \boldsymbol{y}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i + \boldsymbol{\epsilon}_i, \quad \boldsymbol{b}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{D}), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}), \quad i = 1, \ldots, m, \\ \boldsymbol{\epsilon}_i \perp \boldsymbol{\epsilon}_j, \quad \boldsymbol{b}_i \perp \boldsymbol{b}_j, \quad i \neq j; \qquad \boldsymbol{\epsilon}_i \perp \boldsymbol{b}_j, \quad \text{all } i, j \end{gathered} \tag{1}$$

where $\boldsymbol{y}_i$ is the vector of length n_i of responses for subject i; $\boldsymbol{X}_i$ is the $n_i \times p$ model matrix for subject i with respect to $\boldsymbol{\beta}$; and $\boldsymbol{Z}_i$ is the $n_i \times q$ model matrix for subject i and the random effects $\boldsymbol{b}_i$. The symbol $\perp$ indicates independence of random variables. The columns of the model matrices $\boldsymbol{X}_i$ and $\boldsymbol{Z}_i$ are derived from covariates observed for subject i. The $q \times q$ relative dispersion matrix $\boldsymbol{D}$ is symmetric and must be positive definite. Because it contains redundant entries and its elements are constrained we do not express the likelihood in terms of it. Instead we use $\boldsymbol{\theta}$, which is any unconstrained, non-redundant set of parameters that determine $\boldsymbol{D}$.

The maximum likelihood estimates for model (1) are those parameter values, $\widehat{\boldsymbol{\beta}}$, $\widehat{\sigma^2}$ and $\widehat{\boldsymbol{\theta}}$ that maximize

the likelihood or, equivalently, maximize the log-likelihood, of the statistical model given the observed data. The likelihood of the parameters given the data, written $L(\beta, \theta, \sigma^2|y)$, is the same expression as the probability density for the data given the parameters, written $p(y|\beta, \theta, \sigma^2)$, where y is the combined vector of responses for all the subjects. To obtain $p(y|\beta, \theta, \sigma^2)$, we must integrate the conditional density for each subject, $p(y_i|b_i, \beta, \sigma^2)$, with respect to the distribution of the random effects $p(b_i|\theta, \sigma^2)$. Symbolically

$$\begin{aligned} L(\beta, \theta, \sigma^2|y) &= \prod_{i=1}^{m} p(y_i|\beta, \theta, \sigma^2) \\ &= \prod_{i=1}^{m} \int p(y_i|b_i, \beta, \sigma^2)\, p(b_i|\theta, \sigma^2)\, db_i \end{aligned} \tag{2}$$

In a GLMM it is a function $g(\mu)$, called the *link function*, of the mean response that is expressed as the *linear predictor* $X_i\beta + Z_i b_i$. Furthermore, the parameter σ^2 is replaced by a scale parameter and, in the most common types of GLMMs, the scale parameter is constant and is not estimated. Thus we write the model as

$$\begin{gathered} g(\mu_i) = X_i\beta + Z_i b_i \quad b_i \sim \mathcal{N}(0, D), \quad y_i \sim p(\mu_i), \quad i = 1, \ldots, m, \\ y_i \perp y_j, \quad b_i \perp b_j, \quad i \neq j; \qquad \epsilon_i \perp b_j, \quad \text{all } i, j \end{gathered} \tag{3}$$

where $g(\mu)$ is the n_i-dimension vector obtained by applying g componentwise and p is a known distribution such as binomial or Poisson. The likelihood has a form similar to (2)

$$\begin{aligned} L(\beta, \theta|y) &= \prod_{i=1}^{m} p(y_i|\beta, \theta) \\ &= \prod_{i=1}^{m} \int p(y_i|b_i, \beta)\, p(b_i|\theta)\, db_i \end{aligned} \tag{4}$$

with one major difference: the integral in (2) has a closed-form expression but the integral in (4) does not. It is the lack of a closed-form expression for this integral that vastly increases the complexity of parameter estimation for GLMMs relative to LMEs because we must do an iterative optimization of an objective, $L(\beta, \theta)$, that is an integral which must itself be approximated by numerically intensive techniques.

2.2.1 Approximating the integral

The `glmmPQL` approach implemented by Brian Ripley for the `MASS` package has been reimplemented in Phase I of this project. It uses iteratively reweighted LME approximations to the GLMM to jointly optimize

$\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. Because of the LME approximation, an estimate of σ^2 is also produced but not used. More accurate approximations to the likelihood are obtained by approximation the integral in (4). As described in the phase I proposal we will implement the deterministic approximations called second-order Laplacian and adaptive Gauss-Hermite quadrature in Phase II.

As described in Thisted (1988, §5.3), Gaussian quadrature is used to approximate one-dimensional integrals of the form

$$I(a,b) = \int_a^b f(x)w(x)\,dx \tag{5}$$

where $w(x)$ is a weight function. For a fixed weight function and interval, Gaussian quadrature rules can be generated for any order polynomial. Gauss-Hermite quadrature applies to integrals of the form

$$I = \int_{-\infty}^{\infty} f(x)e^{-x^2}\,dx \tag{6}$$

which is approximately the form that we expect the integral $\int p(\boldsymbol{y}_i|\boldsymbol{b}_i,\boldsymbol{\beta})\,p(\boldsymbol{b}_i|\boldsymbol{\theta})\,d\boldsymbol{b}_i$ to take.

In adaptive Gauss-Hermite quadrature we center the integrand $p(\boldsymbol{y}_i|\boldsymbol{b}_i,\boldsymbol{\beta})\,p(\boldsymbol{b}_i|\boldsymbol{\theta})$ about its conditional mode $\widehat{\boldsymbol{b}_i}(\boldsymbol{\beta},\boldsymbol{\theta})$ before applying the Gauss-Hermite rule. This technique requires substantially fewer function evaluations than does Gauss-Hermite quadrature centered at $\boldsymbol{b}_i = \mathbf{0}$ to achieve the same level of accuracy in the approximation. When adaptive Gauss-Hermite quadrature is reduced a single function evaluation it is equivalent to the second-order Laplacian quadrature rule. Second-order and higher-order Laplacian quadrature rules are described in Raudenbush and Bryk (2002).

2.2.2 Determining $\widehat{\boldsymbol{b}_i}(\boldsymbol{\beta},\boldsymbol{\theta})$

The same iteratively reweighted penalized least squares technique used in `glmmPQL` to simultaneously optimize $\boldsymbol{\beta}$ and the $\boldsymbol{b}_i, i = 1,\ldots,m$ can be used for the conditional optimization

$$\widehat{\boldsymbol{b}_i}(\boldsymbol{\beta},\boldsymbol{\theta}) = \arg\max_{\boldsymbol{b}_i} p(\boldsymbol{y}_i|\boldsymbol{b}_i,\boldsymbol{\beta})\,p(\boldsymbol{b}_i|\boldsymbol{\theta}) \tag{7}$$

This greatly simplifies the code for Laplacian and adaptive Gauss-Hermite quadrature.

To demonstrate this, follow the derivation in McCullagh and Nelder (1989, §2.5) including the penalty term.

In providing methods for GLMMs we will restrict our attention to local linearization and to adaptive Gauss-Hermite quadrature. The local linearization technique is called *penalized quasi-likelihood* and has

already been implemented by Prof. Brian Ripley for the MASS package (Venables and Ripley, 1999) for R. The local linear approximation to the GLMM model for PQL is analogous to that used in the IRLS algorithm for GLMs. It replaces the likelihood for the generalized model by a quasi-likelihood for which a least squares estimate can be calculated. The quasi-likelihood is based on working residuals and weights calculated at the current parameter estimates. When the estimates are updated, the weights and working residuals are recalculated, producing a new weighted least squares problem. As described in Pinheiro and Bates (2000, chapter 2) the ML or REML criterion in a linear mixed-effects model can be efficiently evaluated from a penalized least squares problem. The penalized least squares calculation can be applied to GLMMs if the likelihood is replaced by the quasi-likelihood, hence the name penalized quasi-likelihood.

PQL can be regarded as a series of iteratively reweighted linear mixed-effects problems where the optimization of the linear least squares problem uses penalized least squares. In fact, this is exactly Prof. Ripley's implementation. The core of the `glmmPQL` function is a loop that calls `lme` then checks for convergence. If convergence has not been achieved, new weights and working residuals are calculated followed by another call to `lme`. In our experience `glmmPQL` converges quickly, usually after 4 to 8 calls to `lme`. The individual calls to `lme` are relatively fast but could be made much faster. Fitting the 100 simulated data sets from Rodriguez and Goldman (1995) took about 45 minutes in R on a 1.2 GHz desktop computer running Linux. Each of these 100 model fits is a GLMM fit to 2445 observations with two levels of random effects. This is already quite reasonable performance for an interactive language like R but we have been able to reimplement `lme` so as to provide a set of interfaces to the penalized least squares optimization at the core of `lme` that avoid unused, repetitive work in each of these calls to `lme` and achieve greater stability and speed in the algorithm. The stability is achieved because we can use the converged estimates from one iteration as the starting estimates in the next iteration. We have accomplished this reimplementation in Phase I of this project. We have communicated with Prof. Ripley and Dr. Venables regarding this project and they are quite willing to continue to cooperate with us. We will ensure that the R implementation of `glmmPQL` uses the direct interfaces in a reimplemented `lme` whether `glmmPQL` continues to be part of the MASS package or becomes part of the NLME package.

Especially with the changes that we propose for `lme`, `glmmPQL` is a fast and reliable method for estimating parameters in a GLMM. However, unlike GLMs where the IRLS estimates are indeed the maximum likelihood estimates, in GLMMs the PQL estimates are not necessarily the maximum likelihood estimates. For example, the Rodriguez and Goldman (1995) data sets were simulated using fixed-effects parameters $\beta = [0.5, 1., 1., 1.]'$ and with variance components of $\sigma_1 = 1.0$ at the family level and $\sigma_2 = 1.0$ at the com-

munity level. (A GLMM for dichotomous data is based on a binomial distribution and does not have a scale parameter like σ^2.) The average parameter estimates over the 100 simulated data sets fit with `glmmPQL` were $\bar{\hat{\beta}} = [0.6322, 0.9759, 0.9427, 0.9880]'$, $\bar{\hat{\sigma}}_1 = 1.881$, and $\bar{\hat{\sigma}}_2 = 1.2206$. Compared with the estimates from other methods, as given in Rodriguez (2002, Table 10.1), the estimates of the variance components are quite reasonable but the estimates of the fixed-effects parameters, like those from other PQL-based methods, are biased downward.

Systematically underestimating the magnitude of the fixed-effects estimates is a well-known property of the PQL method. Producing better estimates involves using a better approximation to the likelihood — in this case either a Laplacian approximation or adaptive Gauss-Hermite quadrature. Both of these techniques approximate integrals of functions that are close to e^{-x^2} in form. Laplacian integration (Tierney and Kadane, 1986) uses the value and the Hessian of the log-integrand at its conditional mode. Adaptive Gauss-Hermite quadrature uses the value of the log-integrand at the conditional mode and at some number of nearby points whose location is determined from the Hessian of the log-integrand according to the Gauss-Hermite formula. Because the locations of the log-integrand evaluations are symmetric about the conditional mode the total number of points in each axis direction is an odd number. Generally five or seven points will produce sufficient accuracy in AGQ evaluation for GLMMs. A one-point AGQ evaluation is the same as the Laplacian approximation.

Notice that the Laplacian and AGQ methods will require evaluation of the conditional modes of the random effects for each subject every time the likelihood is evaluated during the iterative optimization method. Obviously, the determination of the conditional modes will need to be made as efficient as possible. In our redesign and reimplementation of the `lme` code we will make provision for using IRLS within compiled code for determining the conditional modes of the random effects.

More accurate approximations that use Laplacian approximations or adaptive Gauss-Hermite quadrature are much more compute-intensive than PQL. One approach to parameter estimation is to use the expensive approximations to the log-likelihood throughout the entire iterative process of determining parameter estimates. PROC NLMIXED, available in SAS (http://www.sas.com) versions 7.0 and later, employs this approach using adaptive Gauss-Hermite quadrature throughout. It is an intensive approach and to make it feasible the SAS procedure uses highly tuned, inflexible code. As an example of the inflexibility, PROC NLMIXED cannot be used on the Rodriguez and Goldman simulated data sets because it does not allow for nested random effects.

During an iterative algorithm to optimize the log-likelihood it is not necessary to use an expensive ap-

proximation to the log-likelihood except when close to the optimum. Thus we recommend using simpler approximations initially and, when the simple approximation has converged, switch to a more expensive approximation for final tuning. Because the Laplacian and AGQ evaluations are much more computationally intensive that the PQL iterations, it does not make sense to start with Laplace or AGQ. We propose using PQL initially until it has converged, which, as mentioned above, occurs quite quickly, then switch to Laplace or AGQ to finish the estimation.

In Phase I we concentrated on enhancing the already existing PQL implementation at the level of the penalized least squares problem. The improved performance and interface to the penalized least squares problem will be useful for the Laplacian approximation and the adaptive Gauss-Hermite quadrature approaches in later phases.

2.3 Re-implementation of the `lme` function

Our specific tasks for Phase I included updating the linear algebra calls and interfaces in the R functions `lme`, which fits linear mixed-effects models, and `glmmPQL`, which fits generalized linear mixed models (GLMMs) by penalized quasi-likelihood (PQL). This has been accomplished in the lme4 package for R-1.6.2 (released Jan. 10, 2003).

The classes and methods in lme4 use the formal classes and methods described in Chambers (1998) and implemented in the methods package for R. The underlying numerical methods use Lapack (Anderson et al., 1999) and levels 1, 2, and 3 of the BLAS (Basic Linear Algebra Subroutines). ATLAS (Automatically Tuned Linear Algebra Software) implementations of Lapack and the BLAS are used by R when available. All calls to the underlying C code use the .Call interface (R Development Core Team, 2002, §4.7) that enables us to control the number of copies of objects that are created during the calculations. This can be important when working with large data sets and/or with many terms in a statistical model.

These changes to `lme` and `glmmPQL` have produced a dramatic improvement in performance. In §2.5 below we describe the numerical results of `glmmPQL` applied to 100 sets of simulated multilevel binomial response data. When run on a 2.0 GHz Pentium 4 Linux system the previous version of `glmmPQL` from the MASS package took an average of 30.6 seconds of user time per data set to converge. The version in lme4 took an average of 6.1 seconds of user time per data set.

We have also tested the ability of the new versions to work with large data sets by fitting a multilevel linear model with a fixed-effects parameter vector of length 47 to 378,047 test scores on 134,713 students in

3,722 different schools. Each copy of the model matrix for the fixed effects in this model is approximately 150 MB in size. The new version of `lme` was able to fit this model (on the same 2.0 GHz Pentium 4 system running Linux) in under 5 minutes while keeping the total memory image to less than 1 GB.

2.4 Analytical results for optimization

Another specific task for Phase I is to incorporate the analytic gradient calculations developed by Saikat DebRoy into the `lme` optimization. Doing so provides faster and more reliable convergence in the `lme` function, which is used both for model fitting by itself and in the iteratively reweighted penalized least squares (IRPLS) algorithm used as the inner loop in `glmmPQL` fits of GLMMs. In Phase II IRPLS will be used to determine the conditional modes of the random effects for the Laplacian and the Adaptive Gauss-Hermite Quadrature fits of GLMMs.

We give here an overview of the calculations for ML estimation in a linear mixed-effects model single level of random effects (1).

Calculations are based on a *precision factor* $\boldsymbol{\Delta}$, which is any matrix that satisfies $\boldsymbol{\Delta}'\boldsymbol{\Delta} = \boldsymbol{D}^{-1}$. Given the current $\boldsymbol{\theta}$ we form a series of orthogonal-triangular decompositions

$$\begin{bmatrix} \boldsymbol{Z}_i \\ \boldsymbol{\Delta} \end{bmatrix} = \boldsymbol{Q}_j \begin{bmatrix} \boldsymbol{R}_{11(j)} \\ \boldsymbol{0} \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{R}_{10(j)} \\ \boldsymbol{R}_{00(j)} \end{bmatrix} = \boldsymbol{Q}_j' \begin{bmatrix} \boldsymbol{X}_j \\ \boldsymbol{0} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \boldsymbol{c}_{1(j)} \\ \boldsymbol{c}_{0(j)} \end{bmatrix} = \boldsymbol{Q}_j' \begin{bmatrix} \boldsymbol{y}_j \\ \boldsymbol{0} \end{bmatrix}. \tag{8}$$

followed by

$$\begin{bmatrix} \boldsymbol{R}_{00(1)} & \boldsymbol{c}_{0(1)} \\ \vdots & \vdots \\ \boldsymbol{R}_{00(m)} & \boldsymbol{c}_{0(m)} \end{bmatrix} = \boldsymbol{Q}_0 \begin{bmatrix} \boldsymbol{R}_{00} & \boldsymbol{c}_0 \\ \boldsymbol{0} & c_{-1} \\ \boldsymbol{0} & 0 \end{bmatrix}, \tag{9}$$

These decompositions provide the profiled log-likelihood $\tilde{\ell}(\boldsymbol{\theta})$, the conditional estimates $\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta})$ of the fixed effects, the conditional estimate $\widehat{\sigma^2}(\boldsymbol{\theta})$ of the variance, and the conditional modes $\widehat{\boldsymbol{b}_i}(\boldsymbol{\theta}), i = 1, \ldots, m$ of the

random effects.

$$\tilde{\ell}(\boldsymbol{\theta}) = k - n \log \operatorname{abs} c_{-1} + \sum_{i=1}^{n} \log \operatorname{abs} \frac{|\boldsymbol{\Delta}|}{|\boldsymbol{R}_{11(i)}|} \tag{10}$$

$$\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = \boldsymbol{R}_{00}^{-1} \boldsymbol{c}_0 \tag{11}$$

$$\widehat{\boldsymbol{b}_i}(\boldsymbol{\theta}) = \boldsymbol{R}_{11(i)}^{-1} \left(\boldsymbol{c}_{1(i)} - \boldsymbol{R}_{10(i)} \widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}) \right) \quad i = 1, \ldots, m \tag{12}$$

$$\widehat{\sigma^2}(\boldsymbol{\theta}) = c_{-1}^2 / n \tag{13}$$

With one more orthogonal-triangular decomposition

$$\left[\widehat{\boldsymbol{b}_1}/\widehat{\sigma} \quad \left(\boldsymbol{R}_{11(1)}^{-1} \right) \quad \ldots \quad \widehat{\boldsymbol{b}_m}/\widehat{\sigma} \quad \left(\boldsymbol{R}_{11(m)}^{-1} \right) \right]' = \boldsymbol{U}\boldsymbol{A} \tag{14}$$

we can evaluate the gradient of the profiled log-likelihood as

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = -\frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{q} \left((\boldsymbol{A}'\boldsymbol{A})_{ij} - m\boldsymbol{\Phi}^{ij} \right) \frac{d\boldsymbol{\Phi}_{ij}(\boldsymbol{\theta})}{d\boldsymbol{\theta}} \tag{15}$$

where $(\boldsymbol{A}'\boldsymbol{A})_{ij}$ and $\boldsymbol{\Phi}^{ij}$ are the (i, j)-th elements of $\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{\Phi}^{-1} = \boldsymbol{D}$, respectively.

Optimization of the profiled log-likelihood with respect to $\boldsymbol{\theta}$, even with the analytic gradient (15) available, can be a difficult optimization problem from poorly-chosen initial values. The EM algorithm and the ECME (expectation conditional maximization either) variant of the EM can be used to refine initial estimates. DebRoy and Bates (2003b) derive the ECME update for $\boldsymbol{\Phi}$ as

$$\boldsymbol{\Phi}^{1/2} = \sqrt{m} \left(\boldsymbol{A}^{-1} \right)'$$

The matrix $\boldsymbol{A}$ produced by decomposition (14) contains all the information needed to calculate both the ECME algorithm and the gradient of the profiled log-likelihood, and we use this to produce compact and efficient code for the lme4 package. A single C function called `commonDecompose` is called before an ECME iteration and a gradient calculation. This function calculates and installs the matrix $\boldsymbol{A}$ as the `updateFactor` slot of each `lmeLevel` object and all further calculations can be expressed in terms of that matrix.

The analytic gradients and the ECME iterations are generalized to multiple nested levels of random effects for both maximum likelihood (ML) estimation and restricted maximum likelihood (REML) estimation

Effects	*True value*	*Mean*	*Median*
Fixed effects			
Individual	1.0	0.9584	0.9812
Family	1.0	0.9296	0.9261
Community	1.0	0.9715	0.9589
Random effects (σ)			
Family	1.0	1.6760	1.8494
Community	1.0	1.1794	1.1678

Table 1: Mean and median coefficients for covariates and estimated standard deviations of random effects in the simulated data sets from Rodriguez and Goldman (2001)

in DebRoy and Bates (2003a). All of these results are incorporated into the methods in the lme4 package for R using Lapack and, when available, using ATLAS for maximum efficiency.

2.5 Demonstration using simulated data

We have tested the new implementions of the `lme` and `glmmPQL` functions on several data sets including the simulated data described in Rodriguez and Goldman (1995) and Rodriguez and Goldman (2001). These data are 100 simulated sets of 2445 binary responses grouped into 1558 families in 161 communities. The grouping structure and the values of covariates at each of the levels of community, family, and birth are based on the observed structure of the data from a survey of prenatal care in Guatemala.

The glmmPQL function from the `lme4` package converged on all 100 of the simulated data sets. Summary results are shown in Table 1 and boxplots of the parameter estimates are provided in Figure 1. The summary results can be compared to those for other methods as given in Table 1 of Rodriguez and Goldman (2001).

In evaluating the estimates from the simulated data recall that they will be used as starting estimates for the more accurate adaptive Gauss-Hermite quadrature method in the final version of the GLMM estimation. The objective of the PQL estimation, which is fast and simple, is merely to get into the neighbourhood of the optimum, which these certainly do.

Even these fast, simple estimates are superior to those from many of the methods examined in Rodriguez and Goldman (2001). In the simulation the parameters for which the estimates are shown in Figure 1 were set at 1 then the random noise was added. Estimates for the fixed effects are both accurate (small bias) and precise (small variation). The estimates of σ_1 and σ_2 are biased upward but they are still suitable as starting estimates for AGQ. Although some of the estimates of σ_2 are zero this is not alarming. It is quite possible

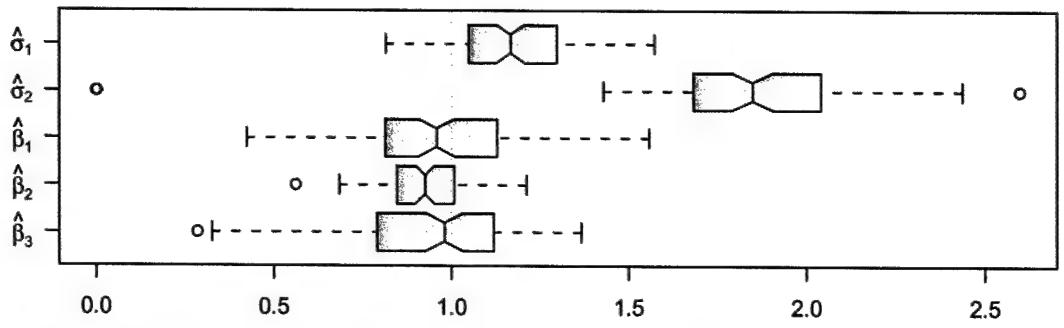

Figure 1: Boxplots of the parameter estimates from `glmmPQL` for the 100 simulated data sets from Rodriguez and Goldman (2001). The parameter σ_1 is the standard deviation of the community-level random effect, σ_2 is the standard deviation of the family-level random effect, β_1 is the coefficient of the community-level covariate, β_2 is the coefficient of the family-level covariate and β_3 is the coefficient of the individual-level covariate. The notches in the boxes represent 95% confidence intervals on the median of the parameter estimates.

for the maximum likelihood estimate of a variance component from simulated data to be zero, even when the data were simulated with a non-zero value. Nevertheless, we will take this into account when using the PQL estimates as starting estimates for AGQ and adjust any zero estimates from PQL before beginning AGQ.

Similar results are obtained using the original `glmmPQL` function from the MASS package, which calls `lme` from the nlme package. The coefficient estimates are not necessarily identical on each of the simulated data sets because Ripley's `glmmPQL` restarts each call to `lme` by calculating initial values whereas the lme4 version of `lme` uses the most recent parameter values. This can affect convergence with the lme4 version converging in fewer iterations on most occasions. As mentioned above, the lme4 version of `glmmPQL` is 5 times as fast as the previous version on these tests.

2.6 Modeling ordinal data

As stated in the project solicitation, the observed data from investigations and studies that could be modeled using this R package are often ordinal in nature. Moreover, the extant software tools are quite limited for modeling ordinal data. Thus, one of the major goals for the R package is to develop the software that can build an accurate GLMM for ordinal data.

As discussed previously, in order to expeditiously build an accurate GLMM for binary data one must proceed rather cautiously; i.e., estimating a GLMM using maximum likelihood methods in a straightforward fashion can result in a large computational effort that is unlikely to produce good parameter estimates. Thus, we propose using a Penalized Quasi-Likelihood (PQL) method to develop an approximate solution be-

fore proceeding to more rigorous computational methods such as Laplacian integration or Adaptive Gauss-Hermite Quadrature (AGQ). Recall that the PQL produces initial estimates for these rigorous methods, and also may be used to quickly investigate several model structures. This initial investigation provides a "first cut" in the model development process, and poor structures can be eliminated from further consideration early on in the process.

Unfortunately, for ordinal data there is currently no analogous procedure to the PQL method. In order to model ordinal data in the R package a different approach is needed. Thus, the main goal for this research is to develop a suitable approach for that could be used for ordinal data in the same fashion that the PQL method is used for binary data.

2.6.1 Ordinal data and the Proportional-Odds Model

The primary type of data for this toolbox are composed of categorical variables, and often these categories have a natural ordering. Such categorical variables with ordered scales are defined as ordinal data (Agresti, 1996). Data sets consisting of ordinal variables are quite commonly found in organizational and sociological studies, for example, performance on a test (poor, fair, or good), attitude toward a proposed policy change (disapprove, neutral, approve, approve strongly), and so on.

Categorical variables without natural ordering are defined as as nominal or multinomial variables. Examples of multinomial data include religious affiliation or the kind of car a person purchases. Multinomial data are also frequently found in the social sciences, and a researcher could use this package to develop mixed-effect models for multinomial data sets. However, the inherent ordering in ordinal data greatly simplifies the modeling process, and was therefore the initial concern for Phase I of this research project.

As discussed previously, a GLMM model extends a GLM to include random effects. Because a GLM for ordinal variables uses the same fundamental approach as a GLM for binary variables, we briefly summarize the approach for binary data here (McCullagh and Nelder, 1989). Let y denote a binary random variable, and the probability of success (indicated by a 1) is denoted by π. In the GLM, a model for π is constructed using a continuous variable, η, which is linearly related to one or more predictors; this continuous variable is related to the categorical variable using a link function, $g(\cdot)$,

$$\eta(\boldsymbol{x}) = \sum_i \beta_i x_i \tag{16}$$

$$g(\pi) = \eta(\boldsymbol{x}) \tag{17}$$

where x_i are the terms used to predict π. Although several link functions may be used, the logit link function is of particular interest for ordinal variables:

$$g(\pi) = \log\left(\frac{\pi}{1-\pi}\right) \tag{18}$$

For an ordinal random variable, suppose there are N categories of interest. Again let y denote the random variable, which may now take on N values corresponding to each category; we label the values for the N categories $0, \ldots, N-1$. Furthermore, let π_j denote the probability of being in category j, and let γ_j denote the cumlative probability for category j; i.e., the probability of being in category 0 to category j. The proportional-odds model for ordinal variables is given by,

$$\log\left(\frac{\gamma_j(\boldsymbol{x})}{1-\gamma_j(\boldsymbol{x})}\right) = \theta_j - \sum_i \beta_i x_i \tag{19}$$

Compared to (18) for binary variables, an additional parameter is introduced, θ_j, which is the threshold for the j^{th} category. The unique feature of a proportional-odds model, and the origin of its name, is that the ratio of the odds for two values of x is given by,

$$\frac{\gamma_j(\boldsymbol{x}_1)/(1-\gamma_j(\boldsymbol{x}_1))}{\gamma_j(\boldsymbol{x}_2)/(1-\gamma_j(\boldsymbol{x}_2))} = e^{-\beta^T(\boldsymbol{x}_1-\boldsymbol{x}_2)} \tag{20}$$

which is independent of the category (j). The proportional-odds model is illustrated in Fig. 2, for a single predictor with $\beta > 0$ and three possible categories for the response variable. Note that the probability for a category is proportional to the area under the curve within the region for that category. Note that as x increases, π_3 increases for $\beta > 0$; if $\beta < 0$, the response probabilities would shift in the opposite direction, and π_1 would increase.

As discussed by Hedeker (2002), random effects may be included in the proportional-odds model in the same way that random effects are added to a GLM for binary data and the logistic regression model. That is, the continuous variable η is predicted with new terms for the random effects,

$$\eta(\boldsymbol{x}, \boldsymbol{z}) = \beta^T \boldsymbol{x} + b^T \boldsymbol{z} \tag{21}$$

where the random effects are assumed to be normally distributed with mean zero and variance σ_b^2. Using the logit link function in (18) and the thresholds for the proportional-odds model in (19), the following

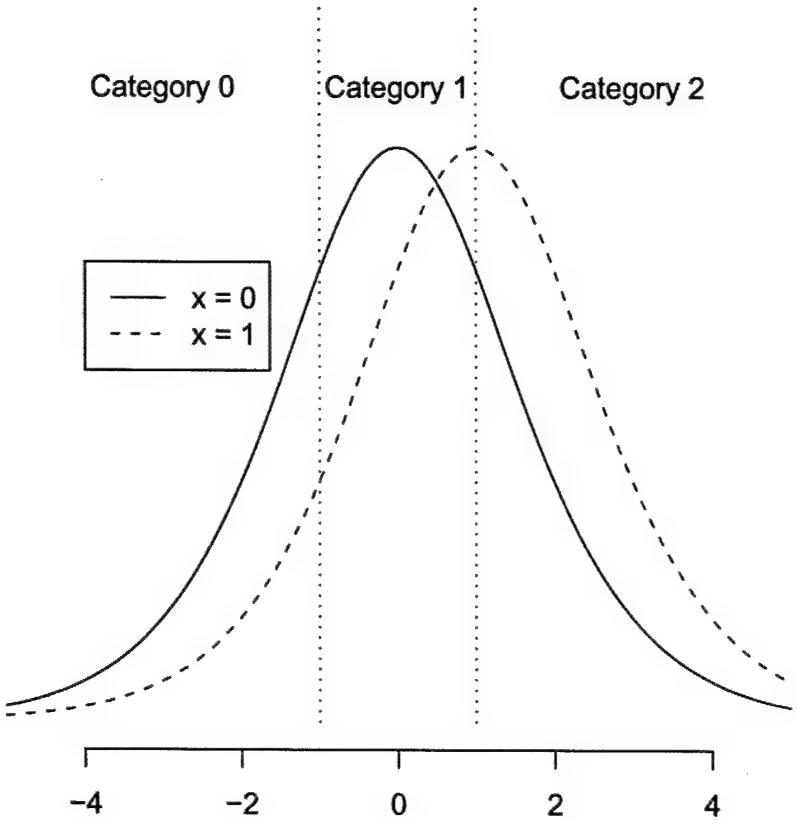

Figure 2: The response probabilities for the proportional-odds model, $\beta > 0$

parameters must be estimated for the mixed-effects, proportional-odds model:

β Fixed-effects parameters

b Conditional modes for the random effects

σ_b^2 Variance of random effects

θ Thresholds for the proportional-odds model

Thus, the same parameters for binary data and the logistic regression model are estimated, and the thresholds, θ, are now estimated for the proportional-odds model.

In principle, these parameters could be estimated directly using MLE. However, as in the case for binary data, the resulting optimization would be very difficult computationally, and unlikely to produce good parameter estimates. Furthermore, the addition of the thresholds precludes the PQL approach from being directly used on the ordinal problem. There is no clear-cut numerical method to use as an approximate procedure for estimating an initial estimate for the MLE and for exploring different model structures during the initial phases of the model development process. Thus, we have investigated two approaches that are somewhat heuristic but should be able to provide an approximate solution.

2.6.2 Continuous Linear Mixed-Effects Modeling

Perhaps the simplest approach for modeling ordinal data would be to ignore the categorical nature of the data and model the data as if it were continuous. That is, if the categorical random variable y falls into one of N categories, the values for y are labelled as $y \in 0, \ldots, N-1$, and these values are modeled as if they were on the continuous scale and did not represent N distinct categories. Thus, the link function in (17) becomes the identity, and a linear, mixed-effects model is used to describe the data,

$$y_c = \beta_c^T \boldsymbol{x} + b_c^T \boldsymbol{z} + e \tag{22}$$

where the subscript "c" is added to denote the continuous-model predictions.

By ignoring the ordinal nature of the data, the linear mixed-effects model in (22) may be fit to the data using a single call to the `lme` function. Building models using this approach is very fast, even for large data sets. Furthermore, as the number of categories increases in y, the data begin to appear "more continuous", and a linear mixed-effects model can be quite accurate.

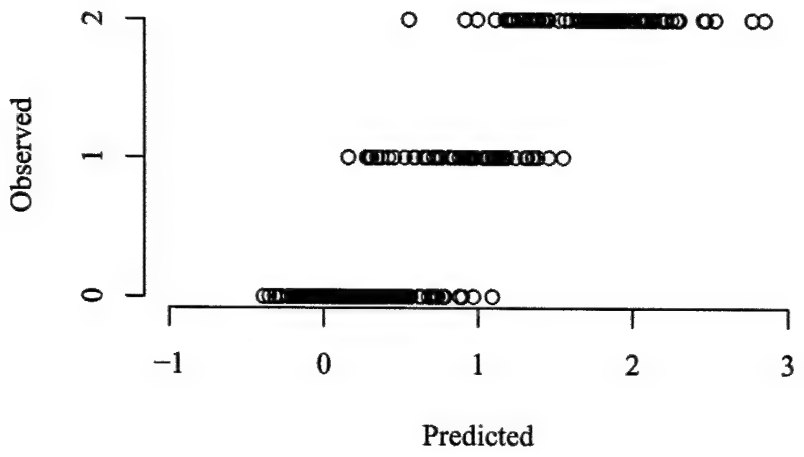

Figure 3: Predictions using the continuous-LME approach.

However, interpreting the model parameters and the model predictions in terms of the true categories becomes difficult. As shown in Fig. 3, the predictions of a linear, mixed-effects model can be quite correlated with the categorical values. But there is no clear meaning for model predictions that lie between categories, or are beyond the range of category values (e.g., $\widehat{y}_c < 0$, or $\widehat{y}_c > 3$). Furthermore, the model parameters on a continuous scale are not related to the proportional-odds model parameters on the categorical scale. Recall that one of the main reasons for approximating the data with a simple model is to provide good initial estimates to a more rigorous method for estimating GLMM parameters.

2.6.3 Dichotomization

An alternative approach is to dichotomize the ordinal data, which transforms the ordinal problem into a binary problem, for which the PQL method may be used to estimate model parameters. Furthermore, by dichotomizing the ordinal data in the way that is outlined below, the model structure for the dichotomized data set directly corresponds to the structure of the proportional-odds model; i.e., the estimated parameters from this step may be used as the initial estimate for a ML method.

Once again, suppose that the ordinal random variable y can be in one of N categories, and these cate-

gories are labeled such that $y \in 0, \ldots, N-1$. The ordinal data set consists of n observations of y, and is denoted as the vector Y. These data may be dichotomized as follows:

1. For each category, calculate $y_{D,j}$, a binary flag that indicates whether y is in categories 0 through j:

$$y_{D,j} = \begin{cases} 0 & y < j \\ 1 & y \geq j \end{cases} \tag{23}$$

   Note that by this definition, $y_{D,0} = 1$.

2. Concatenate the results for $Y_{D,j}$ to create Y_D:

$$Y_D = \begin{bmatrix} Y_{D,1} \\ \vdots \\ Y_{D,N-1} \end{bmatrix} \tag{24}$$

   Note that Y_D will be length $n \times (N-1)$.

3. Define an indicator variable, i_D, which corresponds to an (N-1)-level factor that indicates for which category the current observation of y_D is calculated:

$$i_{D,j} = j + 1 \ \ (j = 1, \ldots, N-1) \tag{25}$$

4. Re-define the model matrix for the fixed-effects to include i_D,

$$\begin{aligned} \tilde{\beta}^T &= [\beta^T \ \beta_D^T] \\ \tilde{x} &= [x \ x_D] \end{aligned}$$

   where x_D is the appropriate model matrix for an (N-1)-level factor (Pinheiro and Bates, 2000), corresponding to i_D:

5. Estimate the mixed-effects model for the binary data, where the formula for η is given by:

$$\eta = \tilde{\beta}^T \tilde{x} + bZ \tag{26}$$

In terms of the proportional-odds model for the ordinal data, the estimates for β, b, and σ_b can be used

directly. Furthermore, the model estimates for β_D, the intercepts for the N-level factor i_D, can be used for the thresholds θ, in the proportional-odds models.

Note that the model predictions can be the inverted back to the ordinal scale. Given values for x and z, the model can predict η on the linear scale; denote this prediction as $\widehat{\eta}$. Using the inverse of the link function (McCullagh and Nelder, 1989), $\widehat{\eta}$ can be converted into $\widehat{\gamma}_j$, predicted cumulative probabilities. The cumulative probabilities can be transformed into the categorical probabilities by:

$$\widehat{\pi}_j = \widehat{\gamma}_j - \widehat{\gamma}_{j-1}$$

The predicted category, $\widehat{y}$, is the category with the largest value of $\widehat{\pi}$.

2.6.4 Proposed Approach

The continuous approach is very fast and might model the ordinal data fairly well (i.e., the model predictions can be quite correlated with the measured values), but the parameters are not clearly recognizable. The dichotomization approach predicts the data as a categorical variable, and produces parameter estimates that can be used as initial estimates for estimating the parameters of the proportional-odds model; however, this method is slower than the continuous approach.

We propose using the continuous method to estimate several models structures, (i.e., different levels and sources of random effects, different predictors, and so on), particularly at the initial stages of the model development process. Using model validation criteria such as the Akaike Information Criterion (AIC) (Burnham and Anderson, 1998), several model strictures can be investigated quickly, and model structures that are clearly inappropriate can be rejected from further consideration.

Using the model structures of interest from this initial analysis, the dichotomization approach can be used to estimate approximate models for the ordinal data. Based on the accuracy of these models, once again using metrics such as the AIC, the candidate set of model structures may be reduced again, and the remaining structures that pass this "second cut" can be estimated using MLE and computationally intensive methods for numerical integration, e.g., Laplacian integration or Adaptive Gauss-Hermite Quadrature. Furthermore, the parameter estimates from this dichotomization method can be used as the initial estimate for the more numerically intensive methods.

2.6.5 Simulation Example

This simulation represents the kind of data that might appear in a typical survey. The response variable is the performance on a test, and may take on three categories: poor, fair, or good, which are enumerated as 0 (poor), 1 (fair), and 2 (good). The predictor variables for the model are *experience*, a continuous variable that measures the years of experience on the job, and *treatment*, a binary variable that corresponds to whether a subject has received a particular kind of training. The data are collected for every soldier within one regiment, and the grouping for mixed-effects corresponds to the nested levels of company, platoon and squad. For the simulation, the data are balanced and there are 4 companies in the regiment, 2 platoons in each company, 2 squads in each platoon, and 10 soldiers (individuals) in each squad.

The response data were simulated using a proportional-odds model. Several proportional-odds models were used to examine different model structures for the random-effects terms only. Denote the random-effects terms as η_{ran} and the fixed-effects terms at η_{fix}, so that the continuous variable, η in (21), can be written as:

$$\eta = \eta_{fix} + \eta_{ran}$$

The fixed-effects for each simulation were:

$$\eta_{fix} = \beta_0 + \beta_{exp} x_{exp} + \beta_{trt} x_{trt} \tag{27}$$

Results were calculated for the following four random effects:

$$\eta_{ran} = \begin{cases} b_i + b_{i,j} + b_{i,j,k} & \text{“Intercept”} \\ b_i + b_{i,j} + b_{i,j,k} + b_{exp,i} & \text{“Experience”} \\ b_i + b_{i,j} + b_{i,j,k} + b_{trt,i} & \text{“Treatment”} \\ b_i + b_{i,j} + b_{i,j,k} + b_{exp,i} + b_{trt,i} & \text{“Both”} \end{cases} \tag{28}$$

where index i corresponds to the company-level effect, j is for the platoon-level effect (i.e., the j^{th} platoon in the i^{th} company, and k is for the squad-level effect. Each of the random-effect coefficients are normally distributed random variables with variance, $\sigma_b^2 = 1$.

The design matrix was generated by specifying x_{exp} as a uniformly distributed random variable between 0 and 5 years, and the treatments were calculated from a binomial distribution, with uniform probability, $p = 0.5$, for all of the individuals. Using 30 Monte Carlo trials, results were calculated for the four simulations

Table 2: Average ΔAIC values for the continuous models.

	Model Structure				
Simulation	None	Intercept	Experience	Treatment	Both
Intercept	231	1.2	4.3	2.0	6.8
Experience	307	69	1.4	71	2.9
Treatment	243	34	37	0.29	5.1
Both	318	72	17	59	0.26

in (28).

Following the proposed approach, the ordinal response data were first modeled with continuous linear mixed-effects models. For each model structure in (28), 30 Monte Carlo trials were generated, and for each trial, five model structures were estimated: four are the exact structures in (28) and the fifth model was a simple linear model (i.e., mixed effects were ignored). For each model, the AIC values were calculated. As discussed by Burnham and Anderson (1998), the AIC metric can be used to select the most appropriate model (or subset of models) from a candidate set of model structures. Furthermore, the AIC is not an absolute measure, and should only be used relative to other models in the candidate set. Thus, for each trial, the ΔAIC was calculated for each model in the candidate set,

$$\Delta\text{AIC}(j) = \text{AIC}(j) - \min \text{AIC}$$

where min AIC is smallest AIC for the candidate set of models. The ΔAIC results for the continuous models are shown in Table 2, avenged over 30 trials. For each trial, ΔAIC = 0 for the selected model; averaged over 30 trials, the best model selection for these simulations has the smallest average ΔAIC.

For each simulation, the smallest average ΔAIC value corresponds to the true model structure. Clearly, mixed-effects models are required as the ΔAIC for models without random effects (the column labelled "None") are quite large. Often the ΔAIC are rather close (e.g., the "Intercept" simulations), indicating that no model is clearly superior; although for the "Both" simulation, the ΔAIC indicates that random effects should be included for the experience and treatment slopes. We should note that these simulations are very fast; thus, using this continuous modeling step, we could quickly identify reasonable model structures for further analysis.

Using the proposed dichotomization method, GLMMs were estimated for the binary data. For each simulation in (28), the `glmmPQL` function was used to estimate a GLMM, using the true model structure. The results from one trial are shown in Table 3, for the "treatment" simulation (i.e., when there are company-

Table 3: Parameter Estimates for "Treatment" Simulation (first Monte Carlo trial)

	Predicted		
Observed	0	1	2
0	156	47	7
1	13	57	48
2	0	3	29

Table 4: Parameter Estimates for "Treatment" Simulation (averaged over 30 trials)

	True	Estimate (Average)	Estimate (Std. Dev.)
β_{exp}	0.1	0.1489	0.0956
β_{trt}	1	1.829	0.939
θ_1	1	1.03	1.01
θ_2	4	4.72	1.14
σ_b(Company,Intercept)	1	1.36	1.25
σ_b(Company,Treatment)	1	1.765	0.999
σ_b(Platoon,Intercept)	1	1.606	0.617
σ_b(Squad,Intercept)	1	1.83	0.37

level effects on the intercept and slope for x_{trt}). The model predictions are relatively accurate, especially for the first and third categories; the middle category gets underpredicted. For all of the simulations in (28), we observed the same behavior, i.e., the model predictions are worst for the middle category.

In Table 4, the true parameters are shown with the estimated parameters' means and standard deviations (for 30 Monte Carlo trials). Note that there is a rather large uncertainty in the parameter estimates, and that the mean values for the estimates are greater than the true values. This behavior also was observed for all of the simulations, and may be symptomatic of the current version of `glmmPQL`, as well as the PQL solution. However, in general the parameters are reasonably close to the true values, and therefore appear to be useful as initial estimates for more refined numerical methods such as Laplacian integration and Gauss-Hermite Quadrature.

2.6.6 Results for an experimental data set

Data collected from a survey of ROTC college students were used to evaluate the proposed dichotomization approach[1]. In the survey, each student was scored according to three metrics related to leadership abilities: performance potential, group conflict, and psychological health. Each of these of metrics could take on three categories (enumerated as 0, 1 and 2). For performance potential and group conflict, the categories

[1]The ROTC data set used in this section was graciously provided by Major Paul Bliese

Table 5: Results for continuous models of performance potential.

	η_{rand}	Δ(AIC)
GPA (Squad)	$b_{\text{GPA},s}x_{\text{GPA},i} + b_s$	0
Gender (Squad)	$b_{\text{Gender},s}x_{\text{Gender},i} + b_s$	2.6
Gender, GPA (Squad)	$b_{\text{Gender},s}x_{\text{Gender},i} + b_{\text{GPA},s}x_{\text{GPA},i} + b_s$	4.6
Intercept (Squad)	b_s	22.5
None	0	330

correspond to binning the students' scores into three classes: the upper third (2), middle third (1), and lower third (0). Psychological health measures the degree of depression: major (0), minor (1), and none (2).

In addition to the leadership metrics, the following data were also collected for each subject: high school grade point average (GPA), gender, efficacy (a measure of self confidence), engagement (a measure of involvement), and adaptability (a measure of "experimental adaptability").

The survey was conducted for eleven regiments and the regiments were grouped according to companies, platoons, and squads (two companies per regiment, four platoons per company, and four squads per platoon). The data set consists of 2,990 observations, although several observations are incomplete. After removing these values, 2,316 observations remain, and the resulting data set is unbalanced – i.e., there are varying numbers of observations for the each level. For small data sets, unbalanced data can be problematic because there may be very few data points for certain levels. Furthermore, even in large data sets where there are several groups, there may be very few data for the lowest level of grouping. In this case, there appear to be sufficient observations for most levels, although the squad-level interactions should be handled with care, and the unbalanced design should not be a problem.

Results for Performance Potential Using Performance Potential as the response variable, we examined several model structures by first modeling the data as if it were continuous. The linear models indicated that all of the predictors are significant, and should be included. In Table 5 the difference in AIC values for several models are shown. Clearly, random effects are needed to model the data accurately. Among the mixed-effects models, the results indicated that effects may have existed at the squad and regiment levels. In Table 5, results are shown only at the squad level for four mixed-effects models. It appears that two-dimensional models may be necessary, i.e., because the Δ(AIC) for models with mixed-effects on the intercept only was 22, which is quite different from models that included random coefficients for the slopes.

Table 6: Parameter estimates for fixed effects, performance potential

	θ_1	θ_2	Gender	GPA	Efficacy	Engagement	Adaptability
PQL	-6.85	-8.42	-.482	0.769	0.303	0.369	0.653
POLR	-6.66	-8.18	-0.481	0.751	0.300	0.383	0.602

Table 7: Parameter estimates for random effects, peformance potential

	σ_{b_r}	σ_{GPA}	σ_{Gender}
Estimate	1.906	0.673	0.928
Component (%)	73	10	17

For the GLMM, the following model structure for the fixed effects was used:

$$\eta_{fix,i} = \boldsymbol{\beta}^T \boldsymbol{X}_i \tag{29}$$

$$\boldsymbol{X}_i = \left[\begin{array}{ccccccc} x_{\text{GPA},i} & x_{\text{Gender},i} & x_{\text{Efficacy},i} & x_{\text{Engagement},i} & x_{\text{Adaptability},i} & I_{1,i} & I_{2,i} \end{array} \right]^T$$

$$\boldsymbol{\beta} = \left[\begin{array}{ccccccc} \beta_{\text{GPA}} & \beta_{\text{Gender}} & \beta_{\text{Efficacy}} & \beta_{\text{Engagement}} & \beta_{\text{Adaptability}} & \theta_1 & \theta_2 \end{array} \right]^T$$

where I_1 and I_2 are the indicator variables and θ_1 and θ_2 are the corresponding thresholds.

In Table 8, the accuracy of the POLR and PQL model predictions is shown, using an "error matrix". The true category is shown in each row of the matrix, and the model predictions are shown in each column. Thus, when the observed category is 0, the PQL model predicts Category 0 for 60%, Category 1 for 24%, and Category 2 for 16% of these observations. Ideally, the error matrix is the identity, indicating that all categories are predicted perfectly. For the PQL model, the diagonal elements of the error matrix are all higher than the corresponding elements for the POLR model's error matrix (i.e., the PQL model is more accurate than the POLR model). One metric for measuring the model accuracy is the "error rate", which equals one minus the average of the diagonal elements. For the PQL model, the error rate is 48% versus the

Table 8: Error matrices for performance potential.

	PQL Predicted			POLR Predicted		
Observed	0	1	2	0	1	2
0	0.60	0.24	0.16	0.53	0.21	0.26
1	0.30	0.31	0.39	0.33	0.24	0.43
2	0.12	0.23	0.65	0.21	0.19	0.60

Table 9: Error matrices for group conflict.

	PQL			POLR		
	Predicted			Predicted		
Observed	0	1	2	0	1	2
0	0.91	0	0.09	0.93	0	0.07
1	0.63	0	0.37	0.93	0	0.07
2	0.28	0	0.72	0.88	0	0.12

PQL model includes squad-level random effects on the intercept and the Adaptability slope.

55% for the POLR model. Thus, the addition of random effects clearly improves the predictability of these data. Note that for both the POLR and PQL results, the middle category (i.e., the middle third of performance potential) is predicted the least accurately; a possible explanation is the thresholds, θ_1 and θ_2 are rather close to one another compared to the spread in the data.

Results for Group Conflict and Psychological Health The same modeling approach was applied using the group conflict and psychological health response variables. Several models with different dimensions and groupings of random effects were examined, with the continuous methods and proposed dichotomization method. For both response variables, only squad-level random effects appeared to be significant. The error matrices for a POLR model and a representative mixed-effects model (estimated using PQL) are shown in Tables 9 and 10, respectively.

For group conflict, neither the POLR or PQL model ever predicts the middle category. This may be the result of two factors: the difficulty of predicting the middle category (i.e., the predictions tend to be skewed toward the outer categories) and the small number of observations in the middle category (24% of the observations are in the middle category, compared to 47% in the first category and 29% in the last category). For psychological health, both models nearly always predict the last category (no signs of depression) and therefore do not accurately model observations in the first or second category. This tendency is most likely due to the lack of data for signs of depression: 82% of the respondents reported no signs of depression, 14% reported minor depression, and only 4% reported major depression.

Clearly, the mixed-effects models estimated with PQL do not predict the categories where there are relatively few data very accurately. In Phase II, we will investigate whether a more accurate mixed-effects model can be estimated using a more numerically intensive method such as Gauss-Hermite Quadrature (i.e., whether such a model accurately predicts categories where there are few observations).

Table 10: Error matrices for psychological health.

	PQL			POLR		
	Predicted			Predicted		
Observed	0	1	2	0	1	2
0	0.01	0.10	0.89	0.01	0.09	0.90
1	0.01	0.02	0.97	0.01	0.03	0.96
2	0	0.01	0.99	0	0.01	0.99

PQL model includes squad-level random effects on the intercept and the Efficacy and Engagement slopes.

3 Key Research Accomplishments

- Re-implemented the `lme` function in the R NLME package.
- Developed analytic results for gradient calculation used in the `lme` function.
- Developed a method for modeling ordinal data approximately.
- Demonstrated the method for ordinal data on simulations and ROTC data set.

4 Reportable Outcomes

- Technical report: "Computational Methods for Single Level Linear Mixed-effects Models" (DebRoy and Bates, 2003b) (Appendix A).
- Technical report: "Computational Methods for Multiple Level Linear Mixed-effects Models" (DebRoy and Bates, 2003a) (Appendix B).
- Presentation: "Converting a large R package to S4 classes and methods", to be presented at the Distributed Statistical Conference, March 20–22, 2003, Vienna Austria (Appendix C).
- We intend to publish a paper on the proposed approach for modeling ordinal data.
- The analytical results will appear in Mr. Saikat Debroy's Ph.D. dissertation (degree expected to be awarded Fall, 2003).

5 Conclusions

A prototype R package for estimating generalized linear mixed-effects models has been developed in Phase I of this research. A key function within the package is the `lme` function. In Phase I, the `lme` function was reimplemented and is now faster and more accurate. Several analytical results were developed that, once implemented in the `lme` function, will also improve the speed and accuracy of the package. Using the proposed dichotomization method, approximate GLMMs may be estimated for ordinal data. The results for a simulation study and real data sets indicated that these approximate models can predict ordinal data reasonably accurately, but tend to be skewed toward the first and last categories and do not accurately predict in categories where there are few observations. A more refined numerical procedure for estimating GLMM

parameters, which will be investigated and developed in Phase II, may be needed to model these situations better.

References

A. Agresti. *An Introduction to Categorical Data Analysis*. John Wiley & Sons, NY, 1996.

E. Anderson, Z. Bai, C. Bishof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *Lapack Users' Guide*. SIAM, 3rd edition, 1999.

K. Burnham and D. Anderson. *Model Selection and Inference*. Springer, NY, 1998.

John M. Chambers. *Programming with Data: A Guide to the S Language*. Springer, New York, 1998.

Jan de Leeuw and Ita Kreft, editors. *Handbook of Quantitative Multilevel Analysis*. Sage, Thousand Oaks, 2002.

Saikat DebRoy and Douglas M. Bates. Computational methods for multiple level linear mixed-effects models. Technical report, Department of Statistics, University of Wisconsin-Madison, 2003a.

Saikat DebRoy and Douglas M. Bates. Computational methods for single level linear mixed-effects models. Technical report, Department of Statistics, University of Wisconsin-Madison, 2003b.

Harvey Goldstein. *Multilevel statistical models*. Halstead Press, New York, 2nd edition, 1995.

Donald Hedeker. *Handbook of Quantitative Multilevel Analysis*, chapter Multilevel models for nominal and ordinal variables. Sage, 2002.

P. McCullagh and J. A. Nelder. *Generalized linear models*. Chapman & Hall Ltd, second edition, 1989. ISBN 0412317605.

José C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-PLUS*. Statistics and Computing. Springer, 2000.

R Development Core Team. *Writing R Extensions*. http://www.r-project.org/doc/manuals/R-exts.pdf, 1.6.1 edition, 2002.

Steven W. Raudenbush and Anthony S. Bryk. *Hierarchical Linear Models*. Sage, Thousand Oaks, 2nd edition, 2002.

Germán Rodriguez. *Handbook of Quantitative Multilevel Analysis*, chapter Multilevel generalized linear models. Sage, 2002.

Germán Rodriguez and Noreen Goldman. An assessment of estimation procedures for multilevel models with binary responses. *Journal of the Royal Statistical Society, Series A, General*, 158:73–89, 1995.

Germán Rodriguez and Noreen Goldman. Improved estimation procedures for multilevel models with binary response: a case study. *J. of the Royal Statist. Soc., Series A*, 164:339–355, 2001.

Tom Snijders and Roel Bosker. *Multilevel Analysis: An introduction to basic and advanced multilevel modeling*. Sage, Thousand Oaks, 1999.

R. A. Thisted. *Elements of Statistical Computing*. Chapman & Hall, London, 1988.

Luke Tierney and Joseph B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81:82–86, 1986.

William N. Venables and Brian D. Ripley. *Modern Applied Statistics with S-PLUS*. Springer, New York, 3rd edition, 1999.

Appendix A: Technical Report

Computational Methods for Single Level Linear Mixed-effects Models

Saikat DebRoy and Douglas Bates*
Department of Statistics
University of Wisconsin – Madison

January 20, 2003

Abstract

Linear mixed-effects models are an important class of statistical models that are used directly in many fields of applications and are also used as iterative steps in fitting other types of mixed-effects models, such as generalized linear mixed models. The parameters in these models are typically estimated by maximum likelihood (ML) or restricted maximum likelihood (REML). In general there is no closed form solution for these estimates and they must be determined by iterative algorithms such as the EM algorithm or Fisher scoring or by general nonlinear optimizers. We recommend using a moderate number of EM iterations followed by general nonlinear optimization of a profiled log-likelihood. In this paper we present a method of calculating analytic gradients of the profiled log-likelihood. This gradient calculation can be implemented very efficiently using matrix decompositions as is done in the nlme packages for R and S-PLUS. Furthermore, the same type of calculation as is used to evaluate the gradient of the profiled log-likelihood can be used to implment an ECME algorithm.

1 Introduction

Linear mixed-effects (LME) models are widely-used statistical models that also are used as iterative steps in fitting other types of mixed-effects models such as non-linear mixed-effects (NLME) models and generalized linear mixed models (GLMMs). Some forms of nonparametric smoothing spline models can also be viewed as LME models (Ke and Wang, 2001).

*This work is supported by U.S. Army Medical Research and Materiel Command under Contract No. DAMD17-02-C-0119. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

The parameters in these models are typically estimated by maximum likelihood (ML) or restricted maximum likelihood (REML). In this paper we consider ML and REML estimation of the parameters in LME models with a single level of random effects. In a companion paper (DebRoy and Bates, 2003) we generalize our results to models with multiple nested levels of random effects.

Laird and Ware (1982) wrote the single-level linear mixed-effects model as

$$\begin{gathered}\boldsymbol{y}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i + \boldsymbol{\varepsilon}_i, \quad \boldsymbol{b}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{\Phi}^{-1}), \quad \boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I}), \quad i = 1, \ldots, m, \\ \boldsymbol{\varepsilon}_i \perp \boldsymbol{\varepsilon}_j, \quad \boldsymbol{b}_i \perp \boldsymbol{b}_j, \quad i \neq j; \qquad \boldsymbol{\varepsilon}_i \perp \boldsymbol{b}_j, \quad \text{all } i, j\end{gathered} \tag{A-1}$$

where $\boldsymbol{y}_i$ is the vector of length n_i of responses for subject i; $\boldsymbol{X}_i$ is the $n_i \times p$ model matrix for subject i and the fixed effects $\boldsymbol{\beta}$; and $\boldsymbol{Z}_i$ is the $n_i \times q$ model matrix for subject i and the random effects $\boldsymbol{b}_i$. The symbol $\perp$ indicates independence of random variables. The columns of the model matrices $\boldsymbol{X}_i$ and $\boldsymbol{Z}_i$ are derived from covariates observed for subject i. The maximum likelihood estimates for model (A-1) are those parameter values that maximize the likelihood or, equivalently, maximize the log-likelihood, of the statistical model given the observed data.

1.1 Log-Likelihood Function

When defining the parameter space for the model (A-1) we must take into account that the matrix $\boldsymbol{\Phi}$ is required to be symmetric and positive definite. Instead of using elements of $\boldsymbol{\Phi}$ as parameters we will use $\boldsymbol{\theta}$, which is any set of non-redundant, unconstrained parameters that determine $\boldsymbol{\Phi}$ (some choices for the mapping $\boldsymbol{\theta} \mapsto \boldsymbol{\Phi}$ are given in Pinheiro and Bates (1996)) and write the log-likelihood for model (A-1) as

$$\begin{aligned}\ell\left(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta} | \boldsymbol{y}\right) = &-\frac{1}{2}\Big[n\log(2\pi\sigma^2) - m\log|\boldsymbol{\Phi}| + \sum_{i=1}^{m}\log|\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi}|\Big] \\ &- \frac{1}{2\sigma^2}\sum_{i=1}^{m}(\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta})'(\boldsymbol{I} - \boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i')(\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta})\end{aligned} \tag{A-2}$$

where $\boldsymbol{y}$ is the concatenation of the $\boldsymbol{y}_i, i = 1, \ldots, m$ and $n = \sum_{i=1}^{m} n_i$ is the total number of observations.

It is straightforward to determine the conditional ML estimates $\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})$ and $\widehat{\sigma^2}_{ML}(\boldsymbol{\theta})$ and the conditional modes of the random effects $\widehat{\boldsymbol{b}}_i(\boldsymbol{\beta}, \boldsymbol{\theta}), i = 1, \ldots, m$ given $\boldsymbol{\theta}$ as

$$\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta}) = \left(\sum_{i=1}^{m}\boldsymbol{X}_i'\boldsymbol{\Sigma}_i^{-1}\boldsymbol{X}_i\right)^{-1}\sum_{i=1}^{m}\boldsymbol{X}_i'\boldsymbol{\Sigma}_i^{-1}\boldsymbol{y}_i \tag{A-3}$$

$$\widehat{\sigma^2}_{ML}(\boldsymbol{\theta}) = \frac{\sum_{i=1}^{m}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)'\boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)}{n} \tag{A-4}$$

$$\widehat{\boldsymbol{b}}_i(\boldsymbol{\beta}, \boldsymbol{\theta}) = (\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'(\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta}) \tag{A-5}$$

where

$$\boldsymbol{\Sigma}_i = \left(I - \boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'\right)^{-1} \tag{A-6}$$

1.2 Log-Restricted-Likelihood Function

The restricted maximum likelihood (REML) estimate can be obtained by maximizing

$$L_R(\sigma^2, \boldsymbol{\theta}|\boldsymbol{y}) = \int L(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{\beta}$$

where $L(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y})$ is the likelihood function for single-level LME model. We are more interested in the log-restricted-likelihood, which is

$$\begin{aligned}\ell_R\left(\sigma^2, \boldsymbol{\theta}|\boldsymbol{y}\right) = -\frac{1}{2}\Big[&(n-p)\log(2\pi\sigma^2) - m\log|\boldsymbol{\Phi}| + \sum_{i=1}^{m}\log|\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi}| \\ &+ \sum_{i=1}^{m}\log|\boldsymbol{X}_i'\boldsymbol{\Sigma}_i^{-1}\boldsymbol{X}_i| \\ &+ \frac{1}{\sigma^2}\sum_{i=1}^{m}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)'\boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)\Big]\end{aligned} \tag{A-7}$$

Because ℓ_R is not a function of $\boldsymbol{\beta}$ there is no "REML estimate" of $\boldsymbol{\beta}$. It is, however, traditional to use the conditional ML estimate (A-3) evaluated at the REML estimate of $\boldsymbol{\theta}$. We can compute the conditional REML estimate of σ^2 as

$$\widehat{\sigma^2}_{REML}(\boldsymbol{\theta}) = \frac{\sum_{i=1}^{m}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)'\boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{y}_i - \boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta})\right)}{n-p} \tag{A-8}$$

1.3 Estimation of $\boldsymbol{\theta}$

Results (A-3) and (A-4) show that we can provide analytic expressions to evaluate the *profiled log-likelihood* $\tilde{\ell}(\boldsymbol{\theta}) = \ell(\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta}), \widehat{\sigma^2}_{ML}(\boldsymbol{\theta}), \boldsymbol{\theta})$, which is a function of $\boldsymbol{\theta}$ only. Using (A-8) we can also calculate the *profiled log-restricted-likelihood* $\tilde{\ell}_R(\boldsymbol{\theta}) = \ell_R(\widehat{\sigma^2}_{REML}(\boldsymbol{\theta}), \boldsymbol{\theta})$. Note that although (A-3) and (A-4) show that we can evaluate a profiled log-likelihood, they are not good computational formulas for doing so. Efficient computational methods for evaluating $\tilde{\ell}(\boldsymbol{\theta})$ and $\tilde{\ell}_R(\boldsymbol{\theta})$ are described in Pinheiro and Bates (2000) and summarized in §2.3.

It is generally much easier to determine the ML (or REML) estimates by optimizing $\tilde{\ell}$ (or $\tilde{\ell}_R$) rather than ℓ (or ℓ_R) when using a general optimization routine because the parameter vector is of smaller dimension - often much smaller. Another way we can make the optimization problem easier is by providing analytic gradients of the profiled log-(restricted)-likelihood. It is possible to use numerical gradients in optimization but analytic gradients are preferred as they

provide more stable and reliable convergence. In section 2 we derive the analytic gradient of the profiled log-(restricted)-likelihood and show how it can be computed accurately and efficiently.

A third way to make the optimization problem easier is to use good starting values. The EM algorithm provides a way of refining starting estimates before beginning general optimization procedures. In section 3 we derive EM and ECME iterations for ML (REML) estimates of model (A-1) and show how these updates may be applied efficiently.

Finally in section 5 we discuss some connections between the gradient result and the ECME iterations.

2 The gradient of the profiled log-likelihood

Because the partial derivatives of ℓ with respect to β and σ^2 at the conditional estimates $\widehat{\beta}_{ML}(\theta)$ and $\widehat{\sigma^2}_{ML}(\theta)$ must be zero, the first two terms in the chain rule expansion of the gradient

$$\frac{d\tilde{\ell}}{d\theta} = \frac{d\beta'}{d\theta}\frac{\partial \ell}{\partial \beta} + \frac{\partial \ell}{\partial \sigma^2}\frac{d\sigma^2}{d\theta} + \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{\frac{\partial \ell}{\partial\{\Phi\}}\right\}_{ij}\frac{d\Phi_{ij}}{d\theta}$$

will vanish, leaving only

$$\frac{d\tilde{\ell}}{d\theta} = \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{\frac{\partial \ell}{\partial\{\Phi\}}\right\}_{ij}\frac{d\Phi_{ij}}{d\theta} \tag{A-9}$$

When evaluating (A-9) we must again take into account the symmetry of Φ. We define a special form of the derivative of a scalar function with respect to a symmetric matrix for this.

2.1 Derivatives for symmetric matrices

Let $f(\Phi)$ be a scalar function of the $q \times q$ symmetric matrix Φ. We will define the symmetric matrix derivative of $f(\Phi)$ with respect to Φ to have elements

$$\left\{\frac{d}{d\{\Phi\}}f(\Phi)\right\}_{ij} = \begin{cases} \frac{1}{2}\frac{d}{d\Phi_{ij}}f(\Phi) & i \neq j \\ \frac{d}{d\Phi_{ij}}f(\Phi) & i = j \end{cases} \tag{A-10}$$

This definition has the property that if Φ is a function of a non-redundant parameter, such as θ, then

$$\frac{df}{d\theta} = \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{\frac{d}{d\{\Phi\}}f(\Phi)\right\}_{ij}\frac{d\Phi_{ij}(\theta)}{d\theta} \tag{A-11}$$

with the sum in (A-11) being over all the elements of Φ, not just the upper or the lower triangle.

Another way of regarding definition (A-10) is to consider $\boldsymbol{Y}$, a general $q \times q$ matrix (i.e. not restricted to being symmetric) that happens to coincide with $\boldsymbol{\Phi}$. Then

$$\frac{d}{d\{\boldsymbol{\Phi}\}} f(\boldsymbol{\Phi}) = \frac{\frac{d}{d\boldsymbol{Y}} f(\boldsymbol{Y}) + \frac{d}{d\boldsymbol{Y}'} f(\boldsymbol{Y})}{2}$$

That is, the derivative $\frac{d}{d\{\boldsymbol{\Phi}\}} f(\boldsymbol{\Phi})$ can be obtained by differentiating the scalar function with respect to $\boldsymbol{\Phi}$ disregarding the symmetry, then symmetrizing the result.

Using this approach and standard results on matrix derivatives (Rogers, 1980) we have the following results for $\boldsymbol{\Phi}$ and $\boldsymbol{A}$ symmetric $q \times q$ matrices and $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ vectors of dimension q.

$$\frac{d}{d\{\boldsymbol{\Phi}\}}(\boldsymbol{\alpha}'\boldsymbol{\Phi}\boldsymbol{\beta}) = \frac{\boldsymbol{\alpha}\boldsymbol{\beta}' + \boldsymbol{\beta}\boldsymbol{\alpha}'}{2} \tag{A-12}$$

$$\frac{d}{d\{\boldsymbol{\Phi}\}}\left(\boldsymbol{\alpha}'(\boldsymbol{A}+\boldsymbol{\Phi})^{-1}\boldsymbol{\beta}\right) = -(\boldsymbol{A}+\boldsymbol{\Phi})^{-1}\frac{\boldsymbol{\alpha}\boldsymbol{\beta}' + \boldsymbol{\beta}\boldsymbol{\alpha}'}{2}(\boldsymbol{A}+\boldsymbol{\Phi})^{-1} \tag{A-13}$$

$$\frac{d}{d\{\boldsymbol{\Phi}\}}\mathrm{tr}\,(\boldsymbol{\Phi}\boldsymbol{A}) = \boldsymbol{A} \tag{A-14}$$

$$\frac{d}{d\{\boldsymbol{\Phi}\}}|\boldsymbol{A}+\boldsymbol{\Phi}| = |\boldsymbol{A}+\boldsymbol{\Phi}|(\boldsymbol{A}+\boldsymbol{\Phi})^{-1} \tag{A-15}$$

$$\frac{d}{d\{\boldsymbol{\Phi}\}}\log|\boldsymbol{A}+\boldsymbol{\Phi}| = (\boldsymbol{A}+\boldsymbol{\Phi})^{-1} \tag{A-16}$$

2.2 Gradient of the profiled log-likelihood

Using the results of the previous section we can express the partial derivative of the log-likelihood(A-2) with respect to $\boldsymbol{\Phi}$ as

$$\begin{aligned}
\frac{\partial \ell}{\partial\{\boldsymbol{\Phi}\}} =& -\frac{1}{2}\left[-m\frac{\partial \log|\boldsymbol{\Phi}|}{\partial\{\boldsymbol{\Phi}\}} + \sum_{i=1}^{m}\frac{\partial \log|\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}|}{\partial\{\boldsymbol{\Phi}\}}\right] \\
&-\frac{1}{2\sigma^2}\sum_{i=1}^{m}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}(\boldsymbol{y}_i-\boldsymbol{X}_i\boldsymbol{\beta})'(I-\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i')(\boldsymbol{y}_i-\boldsymbol{X}_i\boldsymbol{\beta}) \\
=& -\frac{1}{2}\left[-m\boldsymbol{\Phi}^{-1} + \sum_{i=1}^{m}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\right] \\
&-\frac{1}{2\sigma^2}\sum_{i=1}^{m}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'(\boldsymbol{y}_i-\boldsymbol{X}_i\boldsymbol{\beta})(\boldsymbol{y}_i-\boldsymbol{X}_i\boldsymbol{\beta})'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}
\end{aligned}$$

In the profiled log-likelihood the last term can be expressed using the conditional modes (A-5) providing

$$\begin{aligned}\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} &= \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{\frac{d\tilde{\ell}}{d\{\boldsymbol{\Phi}\}}\right\}_{ij}\frac{d\boldsymbol{\Phi}_{ij}}{d\boldsymbol{\theta}} \\ &= \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{-\frac{1}{2}\sum_{i=1}^{m}\left[\frac{\widehat{\boldsymbol{b}_i}\widehat{\boldsymbol{b}_i}'}{\widehat{\sigma^2}} + (\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1} - \boldsymbol{\Phi}^{-1}\right]\right\}_{ij}\frac{d\boldsymbol{\Phi}_{ij}}{d\boldsymbol{\theta}}\end{aligned} \tag{A-17}$$

2.3 Efficient computation of the gradient

Pinheiro and Bates (2000) describe methods for evaluating the profiled log-likelihood using a *relative precision factor* $\boldsymbol{\Delta}$, which is any $q\times q$ matrix satisfying $\boldsymbol{\Delta}'\boldsymbol{\Delta} = \boldsymbol{\Phi}^{-1}$, and a series of QR decompositions of the form

$$\begin{bmatrix}\boldsymbol{Z}_i\\ \boldsymbol{\Delta}\end{bmatrix} = \boldsymbol{Q}_j\begin{bmatrix}\boldsymbol{R}_{11(j)}\\ \boldsymbol{0}\end{bmatrix},\quad \begin{bmatrix}\boldsymbol{R}_{10(j)}\\ \boldsymbol{R}_{00(j)}\end{bmatrix} = \boldsymbol{Q}_j'\begin{bmatrix}\boldsymbol{X}_j\\ \boldsymbol{0}\end{bmatrix}\quad\text{and}\quad \begin{bmatrix}\boldsymbol{c}_{1(j)}\\ \boldsymbol{c}_{0(j)}\end{bmatrix} = \boldsymbol{Q}_j'\begin{bmatrix}\boldsymbol{y}_j\\ \boldsymbol{0}\end{bmatrix}, \tag{A-18}$$

followed by

$$\begin{bmatrix}\boldsymbol{R}_{00(1)} & \boldsymbol{c}_{0(1)}\\ \vdots & \vdots\\ \boldsymbol{R}_{00(m)} & \boldsymbol{c}_{0(m)}\end{bmatrix} = \boldsymbol{Q}_0\begin{bmatrix}\boldsymbol{R}_{00} & \boldsymbol{c}_0\\ \boldsymbol{0} & c_{-1}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}. \tag{A-19}$$

These decompositions provide easily solved triangular systems of equations

$$\boldsymbol{R}_{00}\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = \boldsymbol{c}_0 \tag{A-20}$$

$$\boldsymbol{R}_{11(i)}\widehat{\boldsymbol{b}_i}(\boldsymbol{\theta}) = \boldsymbol{c}_{1(i)} - \boldsymbol{R}_{10(i)}\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta})\quad i = 1,\ldots,m \tag{A-21}$$

for the conditional estimates of $\boldsymbol{\beta}$ and the conditional modes of the random effects. The conditional estimate of σ^2 is $\widehat{\sigma^2}(\boldsymbol{\theta}) = c_{-1}^2/n$ and $(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1} = \boldsymbol{R}_{11(i)}'\boldsymbol{R}_{11(i)}$.

Using these results and taking one more QR decomposition

$$\left[\widehat{\boldsymbol{b}_1}_{ML}/\widehat{\sigma}_{ML}\quad\left(\boldsymbol{R}_{11(1)}^{-1}\right)\quad\ldots\quad\widehat{\boldsymbol{b}_m}_{ML}/\widehat{\sigma}_{ML}\quad\left(\boldsymbol{R}_{11(m)}^{-1}\right)\right]' = \boldsymbol{U}\boldsymbol{A} \tag{A-22}$$

we can evaluate the gradient of the profiled log-likelihood as

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = -\frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{q}\left((\boldsymbol{A}'\boldsymbol{A})_{ij} - m\boldsymbol{\Phi}^{ij}\right)\frac{d}{d\boldsymbol{\theta}}\boldsymbol{\Phi}_{ij}(\boldsymbol{\theta}) \tag{A-23}$$

where $(\boldsymbol{A}'\boldsymbol{A})_{ij}$ and $\boldsymbol{\Phi}^{ij}$ are the (i,j)-th elements of $\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{\Phi}^{-1}$ respectively.

2.4 Gradient of the log-restricted-likelihood

Using the same argument as used for log-likelihood, to compute the gradient of the profiled log-restricted-likelihood, we only need consider the the partial

derivative of the log-restricted-likelihood (A-7) with respect to $\boldsymbol{\Phi}$ which can be computed to be

$$\begin{aligned}
\frac{\partial \ell_R}{\partial\{\boldsymbol{\Phi}\}} &= -\frac{1}{2}\left[-m\frac{\partial \log|\boldsymbol{\Phi}|}{\partial\{\boldsymbol{\Phi}\}} + \sum_{i=1}^{m}\frac{\partial \log|\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}|}{\partial\{\boldsymbol{\Phi}\}}\right] \\
&\quad -\frac{1}{2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\sum_{i=1}^{m}\log|\boldsymbol{X}_i'\left(I-\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'\right)\boldsymbol{X}_i| \\
&\quad -\frac{1}{2\sigma^2}\sum_{i=1}^{m}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\left(\boldsymbol{y}_i-\boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}\right)'\left(I-\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'\right)\left(\boldsymbol{y}_i-\boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML}\right) \\
&= -\frac{1}{2}\left[-m\boldsymbol{\Phi}^{-1}+\sum_{i=1}^{m}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\right] \\
&\quad -\frac{1}{2}\sum_{i=1}^{m}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'\boldsymbol{X}_i\left(\sum_{j=1}^{M}\boldsymbol{X}_j'\boldsymbol{\Sigma}_j^{-1}\boldsymbol{X}_j\right)^{-1}\boldsymbol{X}_i'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1} \\
&\quad -\frac{1}{2\sigma^2}\sum_{i=1}^{m}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}\boldsymbol{Z}_i'(\boldsymbol{y}_i-\boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML})(\boldsymbol{y}_i-\boldsymbol{X}_i\widehat{\boldsymbol{\beta}}_{ML})'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi})^{-1}
\end{aligned}$$

Using (A-5) and the decompositions in §2.3, the profiled log-restricted-likelihood can be written as

$$\begin{aligned}
\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} &= \sum_{i=1}^{q}\sum_{j=1}^{q}\left\{\frac{d\tilde{\ell}_R}{d\{\boldsymbol{\Phi}\}}\right\}_{ij}\frac{d\boldsymbol{\Phi}_{ij}}{d\boldsymbol{\theta}} \\
&= \sum_{i=1}^{q}\sum_{j=1}^{q}\Big\{-\frac{1}{2}\sum_{i=1}^{m}\Big[\frac{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'}{\widehat{\sigma^2}}+\boldsymbol{R}_{11(i)}^{-1}\left(\boldsymbol{R}_{11(i)}'\right)^{-1} \\
&\qquad +\boldsymbol{R}_{11(i)}^{-1}\boldsymbol{R}_{10(i)}\boldsymbol{R}_{00}^{-1}\left(\boldsymbol{R}_{00}'\right)^{-1}\boldsymbol{R}_{10(i)}'\left(\boldsymbol{R}_{11(i)}'\right)^{-1} \\
&\qquad -\boldsymbol{\Phi}^{-1}\Big]\Big\}_{ij}\frac{d\boldsymbol{\Phi}_{ij}}{d\boldsymbol{\theta}}
\end{aligned} \tag{A-24}$$

If we take the QR decomposition

$$\begin{bmatrix}
\widehat{\boldsymbol{b}}_{1\,ML}/\widehat{\sigma}_{REML} \\
\left(\boldsymbol{R}_{11(1)}^{-1}\right)' \\
\left(\boldsymbol{R}_{11(1)}^{-1}\boldsymbol{R}_{10(1)}\boldsymbol{R}_{00}^{-1}\right)' \\
\vdots \\
\widehat{\boldsymbol{b}_m'}_{ML}/\widehat{\sigma}_{REML} \\
\left(\boldsymbol{R}_{11(m)}^{-1}\right)' \\
\left(\boldsymbol{R}_{11(m)}^{-1}\boldsymbol{R}_{10(m)}\boldsymbol{R}_{00}^{-1}\right)'
\end{bmatrix}' = \boldsymbol{U}_R\boldsymbol{A}_R \tag{A-25}$$

we can compute the profiled log-restricted-likelihood as

$$\frac{d\tilde{\ell}_R}{d\boldsymbol{\theta}} = -\frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{q}\left((\boldsymbol{A}_R'\boldsymbol{A}_R)_{ij} - m\boldsymbol{\Phi}^{ij}\right)\frac{d}{d\boldsymbol{\theta}}\boldsymbol{\Phi}_{ij}(\boldsymbol{\theta})$$

3 EM and ECME algorithms for LME models

The EM algorithm (Dempster et al., 1977) is a general iterative algorithm for computing maximum likelihood estimates in the presence of missing or unobserved data. In the case of LME models the typical approach to using the EM algorithm is to consider the $\boldsymbol{b}_i, i = 1, \ldots, m$ as unobserved data. In the terminology of EM algorithm, we call the given data $\boldsymbol{y}_i$ the incomplete data and $\boldsymbol{y}_i$ augmented by $\boldsymbol{b}_i$ the complete data.

The EM algorithm has two steps: in the E step, we compute Q, the expected log-likelihood for the complete data and in the M step we maximize the expected log-likelihood with respect to the parameters in the model.

Liu and Rubin (1994) derived the EM algorithm for LME models using $\boldsymbol{b}_i$ as the missing data. In same paper they also introduce *expectation conditional maximization either* (ECME) algorithms, which are an extension of the EM algorithm. In ECME algorithms the M step is broken down into a number of conditional maximization steps and in each conditional maximization step either the original log-likelihood ℓ or its conditional expectation Q is maximized. The maximization in each step is done by placing constraints on the parameters in such a way that the collection of all the maximization steps is with respect to the full parameter space.

Liu and Rubin (1994) provide an example of an ECME algorithm for LME models. An alternative approach (van Dyk, 2000) uses other candidates for the unobserved data in LME models but we will not pursue that here.

3.1 An EM algorithm for LME models

We first describe the EM algorithm obtained with $\boldsymbol{b}_i$ as the missing data. We denote the current values of the parameters by $\boldsymbol{\beta}_0$, $\boldsymbol{\sigma}_0^2$ and $\boldsymbol{\theta}_0$, which generates $\boldsymbol{\Phi}_0$. These are either starting values or values obtained from the last E and M steps. The parameter estimates to be obtained after an E and an M step are $\boldsymbol{\beta}_1$, $\boldsymbol{\sigma}_1^2$ and $\boldsymbol{\theta}_1$, which generates $\boldsymbol{\Phi}_1$. It happens that in the EM and ECME algorithms we can derive $\boldsymbol{\Phi}_1$ directly without forming $\boldsymbol{\theta}_1$ so we will write formulas in terms of $\boldsymbol{\Phi}_1$.

The log-likelihood for the complete data is

$$\ell\left(\beta, \sigma^2, \Phi | y, b\right) = \sum_{i=1}^{m}\left[-\frac{n_i+q}{2}\log\left(2\pi\sigma^2\right) - \frac{\|y_i - X_i\beta - Z_i b_i\|}{2\sigma^2}\right] + \sum_{i=1}^{m}\left[\frac{\log|\Phi|}{2} - \frac{1}{2\sigma^2}\|\Phi^{1/2} b_i\|^2\right] \quad \text{(A-26)}$$

and the conditional distribution of the random effects is

$$b_i | y_i, \beta_0, \sigma_0^2, \Phi_0 \sim \mathcal{N}\left(\widehat{b}_i(\beta_0, \Phi_0), (Z_i' Z_i + \Phi_0)^{-1}\sigma_0^2\right) \quad \text{(A-27)}$$

where $\widehat{b}_i(\beta_0, \Phi_0)$ is the conditional expected value of b_i (also the conditional mode) given in (A-5).

In the E step we compute $Q(\beta, \sigma^2, \Phi | y, \beta_0, \sigma_0^2, \Phi_0)$, the conditional expectation of $\ell\left(\beta, \sigma^2, \Phi | y, b\right)$ as defined in equation (A-26).

$$\begin{aligned}
&Q(\beta, \sigma^2, \Phi | y, \beta_0, \sigma_0^2, \Phi_0) \\
&= \mathrm{E}_{b|y} \sum_{i=1}^{m}\Big[-\frac{n_i+q}{2}\log(2\pi\sigma^2) - \frac{\|y_i - X_i\beta - Z_i b_i\|^2}{2\sigma^2} \\
&\qquad + \frac{\log|\Phi|}{2} - \frac{\|\Phi^{1/2} b_i\|^2}{2\sigma^2}\Big] \\
&= -\frac{n+mq}{2}\log(2\pi\sigma^2) - \sum_{i=1}^{m}\mathrm{E}_{b_i|y_i}\frac{\|y_i - X_i\beta - Z_i b_i\|^2}{2\sigma^2} \\
&\quad + \frac{m}{2}\log|\Phi| - \sum_{i=1}^{m}\frac{\mathrm{E}_{b_i|y_i}\|\Phi^{1/2} b_i\|^2}{2\sigma^2} \\
&= -\frac{n+mq}{2}\log(2\pi\sigma^2) + \frac{m}{2}\log|\Phi| \\
&\quad - \sum_{i=1}^{m}\left\{\frac{\|\Phi^{1/2}\widehat{b}_i(\beta_0, \Phi_0)\|^2}{2\sigma^2} + \frac{\mathrm{tr}\left[\Phi\sigma_0^2(Z_i' Z_i + \Phi_0)^{-1}\right]}{2\sigma^2}\right\} \\
&\quad - \sum_{i=1}^{m}\left\{\frac{\|y_i - Z_i\widehat{b}_i(\beta_0, \Phi_0) - X_i\beta\|^2}{2\sigma^2} + \frac{\mathrm{tr}\left[Z_i' Z_i\sigma_0^2(Z_i' Z_i + \Phi_0)^{-1}\right]}{2\sigma^2}\right\}
\end{aligned} \quad \text{(A-28)}$$

In the M step, we maximize the expected log-likelihood defined in equation (A-28). To do that, we differentiate $Q(\beta, \sigma^2, \Phi | y, \beta_0, \sigma_0^2, \Phi_0)$ with respect to β, σ^2 and Φ.

$$\frac{\partial Q}{\partial \beta} = -\sum_{i=1}^{m}\frac{y_i - Z_i\widehat{b}_i - X_i\beta}{\sigma^2} \quad \Rightarrow \quad \beta_1 = \left(\sum_{i=1}^{m} X_i' X_i\right)\sum_{i=1}^{m} X_i'\left(y_i - Z_i\widehat{b}_i\right) \quad \text{(A-29)}$$

We use (A-12), (A-14) and (A-15) to compute the matrix derivative of (A-28) with respect to $\boldsymbol{\Phi}$.

$$\begin{aligned}
&\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}Q(\boldsymbol{\beta},\sigma^2,\boldsymbol{\Phi}|\boldsymbol{y},\boldsymbol{\beta}_0,\sigma_0^2,\boldsymbol{\Phi}_0)\\
&\quad=-\frac{1}{2\sigma^2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\left[\sum_{i=1}^{m}\left\{\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i\|^2+\operatorname{tr}\left[\boldsymbol{\Phi}\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]\right\}\right]\\
&\qquad+\frac{m}{2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\log|\boldsymbol{\Phi}| \qquad\qquad (A\text{-}30)\\
&\quad=-\frac{1}{2}\sum_{i=1}^{m}\left[\frac{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'}{\sigma^2}+\frac{\sigma_0^2}{\sigma^2}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}-\boldsymbol{\Phi}^{-1}\right]
\end{aligned}$$

Equating to the zero matrix and substituting σ_1^2 for σ^2

$$\begin{aligned}
&\boldsymbol{\Phi}_1^{-1}=\frac{1}{m\sigma_1^2}\sum_{i=1}^{m}\left\{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'+\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right\}\\
&\Rightarrow\quad\boldsymbol{\Phi}_1=m\sigma_1^2\left[\sum_{i=1}^{m}\left\{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'+\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right\}\right]^{-1} \qquad (A\text{-}31)
\end{aligned}$$

Finally we differentiate (A-28) with respect to σ^2 to get

$$\begin{aligned}
&\frac{\partial}{\partial\sigma^2}Q(\boldsymbol{\beta},\sigma^2,\boldsymbol{\Phi}|\boldsymbol{y},\boldsymbol{\beta}_0,\sigma_0^2,\boldsymbol{\Phi}_0)\\
&\quad=-\frac{n+mq}{2\sigma^2}+\sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i\|^2}{2\sigma^4}+\frac{\operatorname{tr}\left[\boldsymbol{\Phi}\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]}{2\sigma^4}\right\}\\
&\qquad+\sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{y}_i-\boldsymbol{Z}_i\widehat{\boldsymbol{b}}_i-\boldsymbol{X}_i\boldsymbol{\beta}\|^2}{2\sigma^4}+\frac{\operatorname{tr}\left[\boldsymbol{Z}_i'\boldsymbol{Z}_i\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]}{2\sigma^4}\right\}
\end{aligned}$$

Again equating to zero and substituting $\boldsymbol{\beta}_1$ for $\boldsymbol{\beta}$ and $\boldsymbol{\Phi}_1$ for $\boldsymbol{\Phi}$ we get

$$\begin{aligned}
\sigma_1^2=&\sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i\|^2}{n+mq}+\frac{\operatorname{tr}\left[\boldsymbol{\Phi}_1\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]}{n+mq}\right\}\\
&+\sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{y}_i-\boldsymbol{Z}_i\widehat{\boldsymbol{b}}_i-\boldsymbol{X}_i\boldsymbol{\beta}_1\|^2}{n+mq}+\frac{\operatorname{tr}\left[\boldsymbol{Z}_i'\boldsymbol{Z}_i\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]}{n+mq}\right\} \qquad (A\text{-}32)
\end{aligned}$$

We can now substitute the value of Φ_1 from (A-31) in (A-32) to get

$$
\begin{aligned}
(n+mq)\,\sigma_1^2 = &\sum_{i=1}^{m}\left\{m\sigma_1^2\widehat{\boldsymbol{b}}_i'\left(\sum_{j=1}^{M}\{\widehat{\boldsymbol{b}}_j\widehat{\boldsymbol{b}}_j'+\sigma_0^2(\boldsymbol{Z}_j'\boldsymbol{Z}_j+\boldsymbol{\Phi}_0)^{-1}\}\right)^{-1}\widehat{\boldsymbol{b}}_i\right\}\\
&+\sum_{i=1}^{m}\left\{m\sigma_1^2\,\mathrm{tr}\left[\left(\sum_{j=1}^{M}\{\widehat{\boldsymbol{b}}_j\widehat{\boldsymbol{b}}_j'+\sigma_0^2(\boldsymbol{Z}_j'\boldsymbol{Z}_j+\boldsymbol{\Phi}_0)^{-1}\}\right)^{-1}\right.\right.\\
&\qquad\left.\left.\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]\right\}\\
&+\sum_{i=1}^{m}\left\{\|\boldsymbol{y}_i-\boldsymbol{Z}_i\widehat{\boldsymbol{b}}_i-\boldsymbol{X}_i\boldsymbol{\beta}_1\|^2+\mathrm{tr}\left[\boldsymbol{Z}_i'\boldsymbol{Z}_i\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]\right\}
\end{aligned}
$$

Using the fact that $\sum_{i=1}^{m}\mathrm{tr}(A_i)=\mathrm{tr}\,(\sum_{i=1}^{m}A_i)$ we can show that the first two terms together are equal to $mq\sigma_1^2$. Therefore

$$
\sigma_1^2=\frac{1}{n}\sum_{i=1}^{m}\left\{\|\boldsymbol{y}_i-\boldsymbol{Z}_i\widehat{\boldsymbol{b}}_i-\boldsymbol{X}_i\boldsymbol{\beta}_1\|^2+\mathrm{tr}\left[\boldsymbol{Z}_i'\boldsymbol{Z}_i\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]\right\} \qquad \text{(A-33)}
$$

3.2 Efficient computation of Φ_1

To compute Φ_1 we use the same results as in §2.3 except that the conditional means of the random effects are scaled by σ_0, not $\hat{\sigma}(\boldsymbol{\theta})$. That is, the orthogonal-triangular decomposition we use is

$$
\left[\widehat{\boldsymbol{b}}_1/\sigma_0\quad\left(\boldsymbol{R}_{11(1)}^{-1}\right)\quad\ldots\quad\widehat{\boldsymbol{b}}_m/\sigma_0\quad\left(\boldsymbol{R}_{11(m)}^{-1}\right)\right]'=\boldsymbol{U}_1\boldsymbol{A}_1 \qquad \text{(A-34)}
$$

The $q\times q$ upper triangular matrix $\boldsymbol{A}_1$ provides Φ_1 as $\Phi_1^{1/2}=\frac{\sqrt{m}\sigma_1}{\sigma_0}\left(\boldsymbol{A}_1^{-1}\right)'$.

4 An ECME variation to the EM algorithm

Consider the ECME algorithm for estimating the LME model

1. Maximize Q over $\boldsymbol{\Phi}$ by fixing σ^2 to $\widehat{\sigma}_0^2$ and $\boldsymbol{\beta}$ to $\boldsymbol{\beta}_0$ to obtain $\boldsymbol{\Phi}_1$.
2. Maximize ℓ over $\boldsymbol{\beta}$ and σ^2 by fixing $\boldsymbol{\Phi}$ to $\boldsymbol{\Phi}_1$ to obtain $\boldsymbol{\beta}_1$ and $\widehat{\sigma}_1^2$.

Note that in the second maximization step, the constraint is only on $\boldsymbol{\Phi}$. So, $\widehat{\sigma}_1^2$ and $\boldsymbol{\beta}_1$ are same as the ML estimates of σ^2 and $\boldsymbol{\beta}$ given $\boldsymbol{\Phi}=\boldsymbol{\Phi}_1$.

The matrix $\boldsymbol{A}$ we will use to compute this $\boldsymbol{\Phi}_1$ is exactly the triangular factor from (A-22) and

$$
\boldsymbol{\Phi}_1^{1/2}=\frac{\sqrt{m}\widehat{\sigma}_0}{\widehat{\sigma}_0}\left(\boldsymbol{A}_1^{-1}\right)'=\sqrt{m}\left(\boldsymbol{A}_1^{-1}\right)'
$$

By using the ECME algorithm we avoid computing the EM estimate of σ^2 at each step and instead use $\widehat{\sigma}_0^2$, which is much simpler to compute. It is clear

that at the stationary point, the two updates coincide and furthermore this ECME algorithm behaves at least as well as the EM algorithm because using the ML estimate for σ^2 can only increase the value of the log-likelihood. The formal justification for ECME algorithm is a more rigorous proof of this heuristic argument.

4.1 ECME algorithm for REML

To obtain a REML ECME algorithm we consider $\boldsymbol{\beta}$ to be part of the missing data.

$$\begin{aligned}\ell\left(\sigma^2, \boldsymbol{\Phi} | \boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{b}\right) &= \sum_{i=1}^{m}\left[-\frac{n_i+q}{2}\log\left(2\pi\sigma^2\right) - \frac{\|\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta} - \boldsymbol{Z}_i\boldsymbol{b}_i\|}{2\sigma^2}\right] \\ &\quad + \sum_{i=1}^{m}\left[\frac{\log|\boldsymbol{\Phi}|}{2} - \frac{1}{2\sigma^2}\|\boldsymbol{\Phi}^{1/2}\boldsymbol{b}_i\|^2\right] \quad \text{(A-35)}\end{aligned}$$

The joint conditional distribution of $\boldsymbol{\beta}$ and the $\boldsymbol{b}_i$'s is

$$\boldsymbol{b}_i | \boldsymbol{\beta}, \boldsymbol{y}_i, \sigma_0^2, \boldsymbol{\Phi}_0 \sim \mathcal{N}\left(\widehat{\boldsymbol{b}}_i, (\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi}_0)^{-1}\sigma_0^2\right) \quad \text{(A-36)}$$

$$\boldsymbol{\beta} | \boldsymbol{y}_i, \sigma_0^2, \boldsymbol{\Phi}_0 \sim \mathcal{N}\left(\widehat{\boldsymbol{\beta}}_{ML}, \left(\sum_{i=1}^{M}\boldsymbol{X}_i'\boldsymbol{\Sigma}_{i0}^{-1}\boldsymbol{X}_i\right)^{-1}\sigma_0^2\right) \quad \text{(A-37)}$$

where $\boldsymbol{\Sigma}_{i0}$ is defined by (A-6) with $\boldsymbol{\Phi} = \boldsymbol{\Phi}_0$.

For the E step we have to compute $Q_R(\sigma^2, \boldsymbol{\Phi} | \boldsymbol{y}, \sigma_0^2, \boldsymbol{\Phi}_0)$, the conditional expectation of ℓ_R.

$$\begin{aligned}&Q_R(\sigma^2, \boldsymbol{\Phi} | \boldsymbol{y}, \sigma_0^2, \boldsymbol{\Phi}_0) \\ &= E_{\boldsymbol{\beta}|\boldsymbol{y}}\left[E_{\boldsymbol{b}|\boldsymbol{\beta},\boldsymbol{y}}\ell_R(\sigma^2, \boldsymbol{\Phi} | \boldsymbol{y})\right] \\ &= E_{\boldsymbol{\beta}|\boldsymbol{y}}\Bigg[-\frac{n+mq}{2}\log(2\pi\sigma^2) + \frac{m}{2}\log|\boldsymbol{\Phi}| \\ &\qquad - \sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{y}_i - \boldsymbol{Z}_i\widehat{\boldsymbol{b}}_i(\boldsymbol{\beta}, \boldsymbol{\Phi}_0) - \boldsymbol{X}_i\boldsymbol{\beta}\|^2}{2\sigma^2} + \frac{\operatorname{tr}\left[\boldsymbol{Z}_i'\boldsymbol{Z}_i\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi}_0)^{-1}\right]}{2\sigma^2}\right\} \\ &\qquad - \sum_{i=1}^{m}\left\{\frac{\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i(\boldsymbol{\beta}, \boldsymbol{\Phi}_0)\|^2}{2\sigma^2} + \frac{\operatorname{tr}\left[\boldsymbol{\Phi}\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i + \boldsymbol{\Phi}_0)^{-1}\right]}{2\sigma^2}\right\}\Bigg]\end{aligned} \quad \text{(A-38)}$$

We can now specify our ECME algorithm as

1. Maximize Q_R over $\boldsymbol{\Phi}$ by fixing σ^2 to $\widehat{\sigma}_0^2$ to obtain $\boldsymbol{\Phi}_1$.

2. Maximize ℓ_R over σ^2 by fixing $\boldsymbol{\Phi}$ to $\boldsymbol{\Phi}_1$ to obtain $\widehat{\sigma_1^2} = \widehat{\sigma^2}_{REML}$.

In this ECME algorithm we only need to maximize Q_R with respect to $\boldsymbol{\Phi}$. So we only have to compute the expectation of the fifth term in (A-38). Other terms are either free of $\boldsymbol{\Phi}$ or they are constants. Now

$$E_{\beta|\boldsymbol{y}}\left[\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i(\beta,\boldsymbol{\Phi}_0)\|^2\right] = \|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i(\widehat{\beta}_{ML}(\boldsymbol{\Phi}_0),\boldsymbol{\Phi}_0)\|^2$$
$$+\operatorname{tr}\left[\boldsymbol{\Phi}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\boldsymbol{Z}_i'\boldsymbol{X}_i\left(\sum_{j=1}^{M}\boldsymbol{X}_j'\boldsymbol{\Sigma}_{j0}^{-1}\boldsymbol{X}_j\right)^{-1}\boldsymbol{X}_i'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\sigma_0^2\right]$$

So the gradient of Q_R with respect to $\boldsymbol{\Phi}$ is

$$\begin{aligned}
&\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}Q(\sigma^2,\boldsymbol{\Phi}|\boldsymbol{y},\sigma_0^2,\boldsymbol{\Phi}_0)\\
&\quad=-\frac{1}{2\sigma_1^2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\left[\sum_{i=1}^{m}\left\{\|\boldsymbol{\Phi}^{1/2}\widehat{\boldsymbol{b}}_i\|^2+\operatorname{tr}\left[\boldsymbol{\Phi}\sigma_0^2(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]\right\}\right]\\
&\qquad-\frac{1}{2\sigma_1^2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\operatorname{tr}\left[\boldsymbol{\Phi}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\boldsymbol{Z}_i'\boldsymbol{X}_i\left(\sum_{j=1}^{M}\boldsymbol{X}_j'\boldsymbol{\Sigma}_{j0}^{-1}\boldsymbol{X}_j\right)^{-1}\right.\\
&\qquad\qquad\left.\boldsymbol{X}_i'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\sigma_0^2\right]\\
&\qquad+\frac{m}{2}\frac{\partial}{\partial\{\boldsymbol{\Phi}\}}\log|\boldsymbol{\Phi}|\\
&\quad=-\frac{1}{2}\sum_{i=1}^{m}\left[\frac{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'}{\sigma_1^2}+\frac{\sigma_0^2}{\sigma_1^2}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}-\boldsymbol{\Phi}^{-1}\right.\\
&\qquad\qquad+\frac{\sigma_0^2}{\sigma_1^2}(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\boldsymbol{Z}_i'\boldsymbol{X}_i\left(\sum_{j=1}^{M}\boldsymbol{X}_j'\boldsymbol{\Sigma}_{j0}^{-1}\boldsymbol{X}_j\right)^{-1}\\
&\qquad\qquad\left.\boldsymbol{X}_i'\boldsymbol{Z}_i(\boldsymbol{Z}_i'\boldsymbol{Z}_i+\boldsymbol{\Phi}_0)^{-1}\right]
\end{aligned}$$

We equate the above to the zero matrix, use $\sigma_0^2 = \sigma^2 = \widehat{\sigma^2}_{REML}$ and the decomposition (A-25) to get the EM update $\boldsymbol{\Phi}_1^{1/2} = \sqrt{m}\left(\boldsymbol{A}_R^{-1}\right)'$.

5 Discussion

Although not obvious from the derivations it should not come as a surprise that the gradient and of the ECME update for ML criteria both involve the same $q\times q$ matrix $\boldsymbol{A}$. If the EM algorithm has converged so the matrix $\boldsymbol{\Phi}$ is a stationary

point then $\boldsymbol{A} = m\boldsymbol{\Phi}^{-1/2}$ and the gradient of the profiled log-likelihood is zero, as it should be. The same also holds for the REML criteria with $\boldsymbol{A}_R = m\boldsymbol{\Phi}^{-1/2}$ at the stationary point.

The ECME algorithm presented in §4 can be viewed as an attempt to make the gradient zero by setting $\boldsymbol{\Phi}$ to be the matrix that would make the gradient zero if $\boldsymbol{A}$ does not change.

References

Saikat DebRoy and Douglas M. Bates. Computational methods for multiple level linear mixed-effects models. Technical Report 1076, Department of Statistics, University of Wisconsin-Madison, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. 39:1–22, 1977.

Chunlei Ke and Yuedong Wang. Semiparametric nonlinear mixed-effects models and their applications. *Journal of the American Statistical Association*, 96 (456):1272–1298, 2001.

Nan M. Laird and James H. Ware. Random-effects models for longitudinal data. *Biometrics*, 38:963–974, 1982.

Chuanhai Liu and Donald B. Rubin. The ECME algorithm: A simple extension of EM and ECM with faster monotone convergence. *Biometrika*, 81:633–648, 1994.

José C. Pinheiro and Douglas M. Bates. Unconstrained parameterizations for variance-covariance matrices. *Statistics and Computing*, 6:289–296, 1996.

José C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-PLUS*. Statistics and Computing. Springer-Verlag Inc, 2000. ISBN 0-387-98957-9.

Gerald S. Rogers. *Matrix Derivatives*. Marcell Dekker, Inc., New York and Basel, 1980.

David A. van Dyk. Fitting mixed-effects models using efficient EM-type algorithms. *Journal of Computational and Graphical Statistics*, 9(1):78–98, 2000.

Appendix B: Technical Report

Computational Methods for Multiple Level Linear Mixed-effects Models

Saikat DebRoy and Douglas Bates*
Department of Statistics
University of Wisconsin – Madison

February 10, 2003

Abstract

In an earlier paper we provided easily-calculated expressions for the gradient of the profiled log-likelihood and log-restricted-likelihood for single-level mixed-effects models. We also showed how this gradient is related to the update of an ECME (expectation conditional maximization either) algorithm for such single level models. In this paper we extend those results to mixed-effects models with multiple nested levels of random effects.

1 Introduction

In an earlier paper (DebRoy and Bates, 2003) we derived the gradient of the profiled log-likelihood and the log-restricted-likelihood for a linear mixed-effects (LME) model with a single level of random effects. In the Laird-Ware (Laird and Ware, 1982) formulation this model would be written

$$\begin{aligned} \boldsymbol{y}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i + \boldsymbol{\varepsilon}_i, \quad \boldsymbol{b}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{\Phi}^{-1}), \\ \boldsymbol{\varepsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}), \quad i = 1, \ldots, m, \\ \boldsymbol{\varepsilon}_i \perp \boldsymbol{\varepsilon}_j, \quad \boldsymbol{b}_i \perp \boldsymbol{b}_j, \quad i \neq j; \qquad \boldsymbol{\varepsilon}_i \perp \boldsymbol{b}_j, \quad \text{all } i, j \end{aligned} \tag{B-1}$$

*This work is supported by U.S. Army Medical Research and Materiel Command under Contract No. DAMD17-02-C-0119. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

where $\boldsymbol{y}_i$ is the vector of length n_i of responses for subject i; $\boldsymbol{X}_i$ is the $n_i \times p$ model matrix for subject i and the fixed effects $\boldsymbol{\beta}$; and $\boldsymbol{Z}_i$ is the $n_i \times q$ model matrix for subject i and the random effects $\boldsymbol{b}_i$. The symbol $\perp$ indicates independence of random variables. The columns of the model matrices $\boldsymbol{X}_i$ and $\boldsymbol{Z}_i$ are derived from covariates observed for subject i.

The final computational formulas are based on a *relative precision factor* $\boldsymbol{\Delta}$, which is any $q \times q$ matrix satisfying $\boldsymbol{\Delta}'\boldsymbol{\Delta} = \boldsymbol{\Phi}^{-1}$, and a series of QR decompositions of the form

$$\begin{bmatrix} \boldsymbol{Z}_i \\ \boldsymbol{\Delta} \end{bmatrix} = \boldsymbol{Q}_j \begin{bmatrix} \boldsymbol{R}_{11(j)} \\ \boldsymbol{0} \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{R}_{10(j)} \\ \boldsymbol{R}_{00(j)} \end{bmatrix} = \boldsymbol{Q}_j' \begin{bmatrix} \boldsymbol{X}_j \\ \boldsymbol{0} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \boldsymbol{c}_{1(j)} \\ \boldsymbol{c}_{0(j)} \end{bmatrix} = \boldsymbol{Q}_j' \begin{bmatrix} \boldsymbol{y}_j \\ \boldsymbol{0} \end{bmatrix}, \tag{B-2}$$

followed by

$$\begin{bmatrix} \boldsymbol{R}_{00(1)} & \boldsymbol{c}_{0(1)} \\ \vdots & \vdots \\ \boldsymbol{R}_{00(m)} & \boldsymbol{c}_{0(m)} \end{bmatrix} = \boldsymbol{Q}_0 \begin{bmatrix} \boldsymbol{R}_{00} & \boldsymbol{c}_0 \\ \boldsymbol{0} & c_{-1} \\ \boldsymbol{0} & 0 \end{bmatrix}. \tag{B-3}$$

and

$$\left[\widehat{b}_{1ML}/\widehat{\sigma}_{ML} \quad \left(\boldsymbol{R}_{11(1)}^{-1}\right) \quad \ldots \quad \widehat{b}_{mML}/\widehat{\sigma}_{ML} \quad \left(\boldsymbol{R}_{11(m)}^{-1}\right)\right]' = \boldsymbol{U}\boldsymbol{A} \tag{B-4}$$

giving the gradient of the profiled log-likelihood as

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = -\frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{q}\left((\boldsymbol{A}'\boldsymbol{A})_{ij} - m\boldsymbol{\Phi}^{ij}\right)\frac{d}{d\boldsymbol{\theta}}\boldsymbol{\Phi}_{ij}(\boldsymbol{\theta}) \tag{B-5}$$

where $(\boldsymbol{A}'\boldsymbol{A})_{ij}$ and $\boldsymbol{\Phi}^{ij}$ are the (i,j)-th elements of $\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{\Phi}^{-1}$ respectively.

For the log-restricted-likelihood, the objective function optimized by the REML estimates, decomposition (B-4) is replaced by

$$\begin{bmatrix} \widehat{b}_{1ML}'/\widehat{\sigma}_{REML} \\ \left(\boldsymbol{R}_{11(1)}^{-1}\right)' \\ \left(\boldsymbol{R}_{11(1)}^{-1}\boldsymbol{R}_{10(1)}\boldsymbol{R}_{00}^{-1}\right)' \\ \vdots \\ \widehat{b}_{mML}'/\widehat{\sigma}_{REML} \\ \left(\boldsymbol{R}_{11(m)}^{-1}\right)' \\ \left(\boldsymbol{R}_{11(m)}^{-1}\boldsymbol{R}_{10(m)}\boldsymbol{R}_{00}^{-1}\right)' \end{bmatrix}' = \boldsymbol{U}_R\boldsymbol{A}_R \tag{B-6}$$

Furthermore an ECME update of $\boldsymbol{\Delta}$ can be calculated as

$$\boldsymbol{\Phi}_1^{1/2} = \frac{\sqrt{m}\widehat{\sigma}_0}{\widehat{\sigma}_0}\left(\boldsymbol{A}_1^{-1}\right)' = \sqrt{m}\left(\boldsymbol{A}_1^{-1}\right)'$$

for maximum likelihood (ML) estimation or the same expression with $\boldsymbol{A}_R$ in place of $\boldsymbol{A}$ for REML estimation.

In this paper we generalize these results to linear mixed-effects models with multiple nested levels of random effects.

1.1 Multilevel LME models

A general linear mixed-effects model would be written

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b} + \boldsymbol{\varepsilon} \tag{B-7}$$

where $\boldsymbol{y}$ is the response, $\boldsymbol{X}$ and $\boldsymbol{Z}$ are fixed effects and random effects model matrices, $\boldsymbol{\beta}$ is the vector of fixed effects parameters, $\boldsymbol{b}$ is the random effects vector and $\boldsymbol{\varepsilon}$ is the error vector and both $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$ are assumed to have Gaussian distributions.

We will consider specific forms of (B-7) that generalize (B-1) to multiple nested levels of random effects. Such models are sometimes called *multilevel models* (Goldstein, 1987) or *hierarchical linear models* (Raudenbush and Bryk, 2002). Let Q be the number of nested levels of random effects. We will number the levels so that the Q-th level is innermost. That is, the level-1 random effects apply to the largest groups of experimental units, the level-2 random effects to the next largest groups nested within the largest groups, and so on, up to level Q. The fixed effects $\boldsymbol{\beta}$ apply to all the observations which could be considered as a single group of observations at a hypothetical zeroth level of grouping.

Two typical examples of nested classifications of experimental units resulting in such multilevel data are students within classes within schools within school districts in an educational system or soldiers within squads within platoons within companies within regiments in an army. In the educational system we would call the school districts level 1, the schools level 2, and so on. In the multilevel modelling literature the levels are often numbered in the opposite order; from innermost to outermost.

We will assume that the observations are ordered so that observations in the same group are adjacent for all Q levels of nested classification. At the i-th level of classification we will number the groups from 1 to m_i, the total number of groups at that level and write indices according to the group

number and the level. That is, we will write the level i response vectors as $\boldsymbol{y}_{i(j)}$ of length $n_{i(j)}, j = 1, \ldots, m_i$. We will extend this notation to the hypothetical level 0 for which $m_0 = 1$ and $n_{0(1)} = n$, the total number of observations.

The model matrix for the fixed-effects is $\boldsymbol{X}$ of size $n \times p$. When discussing a particular level of groups we will refer to the submatrices consisting of the rows of $\boldsymbol{X}$ for group j at level i as $\boldsymbol{X}_{i(j)}$ of size $n_{i(j)} \times p$. Recall that "level 0" corresponds to the fixed-effects which apply to all n observations so that $\boldsymbol{y} = \boldsymbol{y}_{0(1)}$ and $\boldsymbol{X} = \boldsymbol{X}_{0(1)}$.

The random effects at level k, $\boldsymbol{b}_{k(j)}, j = 1, \ldots, m_k$ are each of dimension q_k. The corresponding model matrices are of size $n_{k(j)} \times q_k$. At times we will need to refer to the rows of the model matrix for the level k random effects corresponding to group j at level i, which we designate as $\boldsymbol{Z}_{ki(j)}$. The model matrix $\boldsymbol{Z}$ for the complete set of random effects $\boldsymbol{b} = \left(\boldsymbol{b}_Q', \ldots, \boldsymbol{b}_1'\right)'$ where $\boldsymbol{b}_i = \left(\boldsymbol{b}_{i(1)}', \ldots, \boldsymbol{b}_{i(m_i)}'\right)$ is

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_Q & \ldots & \boldsymbol{Z}_1 \end{bmatrix}$$

where each $\boldsymbol{Z}_i$ is defined to be

$$\boldsymbol{Z}_i = \begin{bmatrix} \boldsymbol{Z}_{ii(1)} & 0 & 0 & \ldots & 0 \\ 0 & \boldsymbol{Z}_{ii(2)} & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & \boldsymbol{Z}_{ii(m_i)} \end{bmatrix}, \quad i = 1, \ldots, Q$$

Notice that the model matrix $\boldsymbol{Z}$ can be completely defined from components of the form $\boldsymbol{Z}_{ii(j)}, i = 1, \ldots, Q$, $j = 1, \ldots, m_i$. The notation $\boldsymbol{Z}_{ki(j)}$ for $k \neq i$ is used simply to designate the rows corresponding to group j at level i of the model matrix for the level k random effects.

The model specification is completed by specifying the distribution of the random effects and the random noise $\boldsymbol{\varepsilon}$

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}) \qquad \boldsymbol{b}_{k(i)} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{\Phi}_i^{-1})$$

$$\boldsymbol{b}_{k(i)} \perp \boldsymbol{b}_{k(j)},\ i \neq j \text{ all } k, \quad \boldsymbol{\varepsilon} \perp \boldsymbol{b}_{k(i)},\ \text{all } i, k \quad \boldsymbol{b}_{k(i)} \perp \boldsymbol{b}_{k'(j)}\ k \neq k', \text{all } i, j \tag{B-8}$$

where $\perp$ indicates independence of two random variables. The relative dispersion matrices $\boldsymbol{\Phi}_i, i = 1, \ldots, Q$ are gathered as

$$\boldsymbol{\Phi}_A = \bigoplus_{i=1}^{Q} (\boldsymbol{I}_{m_{Q-i+1}} \otimes \boldsymbol{\Phi}_{Q-i+1})$$

These matrices must be positive definite, symmetric matrices. We use $\boldsymbol{\theta}$, to represent a non-redundant, unconstrained parametrization for $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_Q$.

The log-likelihood for $\boldsymbol{y}$ is given by

$$\ell\left(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta} | \boldsymbol{y}\right) = -\frac{1}{2}\Big[n \log(2\pi\sigma^2) + \log\left|\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{\Phi}_A^{-1}\boldsymbol{Z}'\right| + \frac{1}{\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'\left(\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{\Phi}_A^{-1}\boldsymbol{Z}'\right)^{-1}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\Big]$$

Using the identities

$$\left|\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{\Phi}_A^{-1}\boldsymbol{Z}'\right| = \frac{\left|\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A\right|}{\left|\boldsymbol{\Phi}_A\right|}$$

$$\left(\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{\Phi}_A^{-1}\boldsymbol{Z}'\right)^{-1} = I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'$$

we can rewrite the log-likelihood as

$$\ell\left(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta} | \boldsymbol{y}\right) = -\frac{1}{2}\Big[n \log(2\pi\sigma^2) + \log\left|\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A\right| - \log\left|\boldsymbol{\Phi}_A\right| + \frac{1}{\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\Big] \quad \text{(B-9)}$$

For a given value of $\boldsymbol{\theta}$, the conditional ML estimates of $\boldsymbol{\beta}$ and σ^2 are given by

$$\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}_{\boldsymbol{X}}\boldsymbol{X}'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)\boldsymbol{y} \quad \text{(B-10)}$$

$$\widehat{\sigma^2}_{ML}(\boldsymbol{\theta}) = \frac{1}{n}\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{ML}\right)'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{ML}\right) \quad \text{(B-11)}$$

where

$$\boldsymbol{\Sigma}_{\boldsymbol{X}} = \left(\boldsymbol{X}'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)^{-1}\boldsymbol{X}\right)^{-1} \quad \text{(B-12)}$$

and the log-restricted-likelihood is given by

$$\ell_R\left(\sigma^2, \boldsymbol{\theta} | \boldsymbol{y}\right) = -\frac{1}{2}\Big[(n-p)\log(2\pi\sigma^2) + \log\left|\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A\right| - \log\left|\boldsymbol{\Phi}_A\right| + \log\left|\boldsymbol{X}'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)\boldsymbol{X}\right| + \frac{1}{\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\Big] \quad \text{(B-13)}$$

Because ℓ_R is not a function of $\boldsymbol{\beta}$ there is no conditional "REML estimate" of $\boldsymbol{\beta}$. (It is, however, traditional to use the conditional ML estimate

(B-10) evaluated at the REML estimate of $\boldsymbol{\theta}$ as the final estimate of $\boldsymbol{\beta}$.) We can compute the conditional REML estimate of σ^2 as

$$\widehat{\sigma^2}_{REML}(\boldsymbol{\theta}) = \frac{\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{ML}\right)'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{ML}\right)}{n - p} \tag{B-14}$$

As we can obtain analytical expressions for the ML estimate of $\boldsymbol{\beta}$ and the conditional estimates of σ^2, we can optimize with respect to $\boldsymbol{\theta}$ only. That is, we optimized the *profiled log-likelihood* $\tilde{\ell}(\boldsymbol{\theta}) = \ell(\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta}), \widehat{\sigma^2}_{ML}(\boldsymbol{\theta}), \boldsymbol{\theta})$ or the *profiled log-restricted-likelihood* $\tilde{\ell}_R(\boldsymbol{\theta}) = \ell(\widehat{\boldsymbol{\beta}}_{ML}(\boldsymbol{\theta}), \sigma^2_{REML}(\boldsymbol{\theta}), \boldsymbol{\theta})$.

1.2 Efficient computation of the log-likelihood

The decomposition methods (B-2) and (B-3) for evaluating the profiled log-likelihood or log-restricted-likelihood were described for both single- and multiple-level LME models in Pinheiro and Bates (2000, chap. 2). In DebRoy and Bates (2003) we showed that these decompositions followed by (B-4) can be used to calculate the gradient of the profiled objective function and to provide an ECME update. Here we generalize the calculation of the profiled log-likelihood or log-restricted-likelihood to multiple levels of random effects. Later we will show how the generalization of the gradient and ECME calculations.

Consider the Cholesky factorization

$$\begin{bmatrix} \boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A & \boldsymbol{Z}'\boldsymbol{X} & \boldsymbol{Z}'\boldsymbol{y} \\ \boldsymbol{X}'\boldsymbol{Z} & \boldsymbol{X}'\boldsymbol{X} & \boldsymbol{X}'\boldsymbol{y} \\ \boldsymbol{y}'\boldsymbol{Z} & \boldsymbol{y}'\boldsymbol{X} & \boldsymbol{y}'\boldsymbol{y} \end{bmatrix} = \boldsymbol{R}'\boldsymbol{R}$$

where $\boldsymbol{R}$ is upper triangular. Because of the structure of $\boldsymbol{Z}$ and $\boldsymbol{\Phi}_A$ we can partition $\boldsymbol{R}$ as

$$\begin{bmatrix} \boldsymbol{R}_{QQ} & \dots & \boldsymbol{R}_{Q2} & \boldsymbol{R}_{Q1} & \boldsymbol{R}_{Q0} & \boldsymbol{c}_Q \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \dots & \boldsymbol{R}_{22} & \boldsymbol{R}_{21} & \boldsymbol{R}_{20} & \boldsymbol{c}_2 \\ \boldsymbol{0} & \dots & \boldsymbol{0} & \boldsymbol{R}_{11} & \boldsymbol{R}_{10} & \boldsymbol{c}_1 \\ \boldsymbol{0} & \dots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{R}_{00} & \boldsymbol{c}_0 \\ \boldsymbol{0} & \dots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{c}_{-1} \end{bmatrix}$$

where for $i \geq 1$ $\boldsymbol{R}_{ii} = \text{diag}(\boldsymbol{R}_{ii(1)}, \dots, \boldsymbol{R}_{ii(m_i)}$, each $\boldsymbol{R}_{ii(j)}$ is upper triangular of size $q_i \times q_i$. Because we have ordered the observations so that the groups

are in adjacent rows, we can ensure that for $i \geq 1, j \geq 0$

$$\boldsymbol{R}_{ij} = \begin{bmatrix} \boldsymbol{R}_{ij(1,1)} & \mathbf{0} & \cdots & \mathbf{0} \\ \ddots & \vdots & \vdots & \vdots \\ \boldsymbol{R}_{ij(m_{ij(1)},1)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{R}_{ij(m_{ij(1)}+1,2)} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \boldsymbol{R}_{ij(m_{ij(1)}+m_{ij(2)},2)} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \cdots & \boldsymbol{R}_{ij(m_i-m_{ij(m_j)}+1,m_j)} \\ \vdots & \vdots & \vdots & \ddots \\ \mathbf{0} & \cdots & \cdots & \boldsymbol{R}_{ij(m_i,m_j)} \end{bmatrix},$$

where $\boldsymbol{R}_{ij(kl)}$ is a general matrix of size $q_i \times q_j$ and $m_{ij(k)}$ is the number of level-i groups which are nested within the k-th level-j group. Also, for $i \geq 1$

$$\boldsymbol{c}_i = \begin{bmatrix} \boldsymbol{c}_{i(1)} \\ \vdots \\ \boldsymbol{c}_{i(m_i)} \end{bmatrix}, \quad \boldsymbol{R}_{i0} = \begin{bmatrix} \boldsymbol{R}_{i0(1)} \\ \vdots \\ \boldsymbol{R}_{i0(m_i)} \end{bmatrix}$$

where $\boldsymbol{c}_{i(j)}$ is of size q_i and $\boldsymbol{R}_{i0(j)}$ is of size $q_i \times p$.

Instead of computing the $\boldsymbol{R}$ matrix from a Cholesky factorization, we use a series of QR decompositions to obtain $R_{ii(k)}$ and $R_{ij(kl)}$. When $Q = 1$ these decompositions are exactly (B-2). The decompositions for $Q = 2$ are shown in detail in Pinheiro and Bates (2000, chap. 2).

In terms of these decomposed matrices the profiled log-likelihood and log-restricted-likelihood are

$$\tilde{\ell}(\boldsymbol{\theta}|\boldsymbol{y}) = const - n\|\boldsymbol{c}_{-1}\| + \sum_{i=1}^{Q}\sum_{k=1}^{m_i}\left(\log|\boldsymbol{\Delta}_i| - \log|\boldsymbol{R}_{ii(k)}|\right)$$

$$\tilde{\ell}_R(\boldsymbol{\theta}|\boldsymbol{y}) = const - (n-p)\|\boldsymbol{c}_{-1}\| - \log|\boldsymbol{R}_{00}| + \sum_{i=1}^{Q}\sum_{k=1}^{m_i}\left(\log|\boldsymbol{\Delta}_i| - \log|\boldsymbol{R}_{ii(k)}|\right)$$

2 Gradient of the profiled log-likelihood

Expression (B-5) for single level LME models is derived in DebRoy and Bates (2003) For multiple levels this generalizes to

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = \sum_{k=1}^{Q}\sum_{i=1}^{q_k}\sum_{j=1}^{q_k}\left\{\frac{\partial \ell}{\partial\{\boldsymbol{\Phi}_k\}}\right\}_{ij}\frac{d\{\boldsymbol{\Phi}_k\}_{ij}}{d\boldsymbol{\theta}} \tag{B-15}$$

where $\frac{\partial}{\partial\{\boldsymbol{\Phi}_k\}}$ denotes the partial derivative of ℓ with respect to the symmetric matrix $\boldsymbol{\Phi}_k$, as defined in DebRoy and Bates (2003).

We can also write it as $\sum_{k=1}^{Q}\frac{\partial \ell}{\partial\{\boldsymbol{\Phi}_k\}} * \frac{d\boldsymbol{\Phi}_k}{d\boldsymbol{\theta}}$ where the $m \times n$ matrix $\boldsymbol{A} * \boldsymbol{B}$ is the star product (Rogers, 1980) defined by

$$\{\boldsymbol{A} * \boldsymbol{B}\}_{i,j} = \sum_{k=1}^{p}\sum_{l=1}^{q}\boldsymbol{A}_{k,l}\boldsymbol{B}_{(k-1)m+i,(l-1)n+j}$$

with $\boldsymbol{A}$ of size $p \times q$ and $\boldsymbol{B}$ of size $mp \times nq$,

The partial derivative of (B-9) with respect to $\boldsymbol{\Phi}_A$ is given by

$$\begin{aligned}
&\frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}}\ell\left(\beta, \sigma^2, \boldsymbol{\Phi}_A|\boldsymbol{y}\right) \\
&\qquad = -\frac{1}{2}\left[\frac{\partial \log|\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A|}{\partial\{\boldsymbol{\Phi}_A\}} - \frac{\partial \log|\boldsymbol{\Phi}_A|}{\partial\{\boldsymbol{\Phi}_A\}}\right] \\
&\qquad\quad - \frac{1}{2\sigma^2}\frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'\left(I - \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\right)(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \\
&\qquad = -\frac{1}{2}\left[\left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A\right)^{-1} - \boldsymbol{\Phi}_A^{-1}\right] \\
&\qquad\quad - \frac{1}{2\sigma^2}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}\boldsymbol{Z}'\boldsymbol{X}\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right) \\
&\qquad\qquad \left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)'\boldsymbol{X}'\boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1} \\
&\qquad = -\frac{1}{2}\left[\frac{\widehat{\boldsymbol{b}}\widehat{\boldsymbol{b}}'}{\sigma^2} + \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A\right)^{-1} - \boldsymbol{\Phi}_A^{-1}\right]
\end{aligned}$$

In order to compute the derivative with respect to $\boldsymbol{\Phi}_i$ we need

$$\begin{aligned}\frac{\partial \boldsymbol{\Phi}_A}{\partial\{\boldsymbol{\Phi}_i\}} &= \frac{\partial}{\partial\{\boldsymbol{\Phi}_i\}}\bigoplus_{k=1}^{Q}(\boldsymbol{I}_{m_{Q-k+1}} \otimes \boldsymbol{\Phi}_{Q-k+1}) \\ &= \bigoplus_{k=1}^{Q}\frac{\partial}{\partial\{\boldsymbol{\Phi}_i\}}(\boldsymbol{I}_{m_{Q-k+1}} \otimes \boldsymbol{\Phi}_{Q-k+1}) \qquad \text{(B-16)} \\ &= \bigoplus_{k=1}^{Q}(\boldsymbol{I}_{m_{Q-k+1}} \otimes \frac{\partial \boldsymbol{\Phi}_{Q-k+1}}{\partial\{\boldsymbol{\Phi}_i\}})\end{aligned}$$

where $\frac{\partial \boldsymbol{\Phi}_k}{\partial\{\boldsymbol{\Phi}_i\}} = \mathbf{0}$ if $i \neq k$ and $\frac{\partial \boldsymbol{\Phi}_i}{\partial\{\boldsymbol{\Phi}_i\}} = \text{vec}(\boldsymbol{I}_{q_i})\,\text{vec}'(\boldsymbol{I}_{q_i})$.

Let $(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{ii}$ be the block in $(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}$ that corresponds to the $\boldsymbol{Z}_i'\boldsymbol{Z}_i$ block in $\boldsymbol{Z}'\boldsymbol{Z}$ and $(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{ii(k)}$ be the block in $(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1}$ that corresponds to the $\boldsymbol{Z}_{ii(k)}'\boldsymbol{Z}_{ii(k)}$ block in $\boldsymbol{Z}'\boldsymbol{Z}$.

Using (B-16),

$$\begin{aligned}\frac{\partial \ell}{\partial\{\boldsymbol{\Phi}_i\}} &= \frac{\partial \ell}{\partial\{\boldsymbol{\Phi}_A\}} * \frac{\partial \boldsymbol{\Phi}_A}{\partial\{\boldsymbol{\Phi}_i\}} \\ &= -\frac{1}{2}\left[\frac{\widehat{\boldsymbol{b}}\widehat{\boldsymbol{b}}'}{\sigma^2} + (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{-1} - \boldsymbol{\Phi}_A^{-1}\right] \\ &\quad * \bigoplus_{k=1}^{Q}(\boldsymbol{I}_{m_{Q-k+1}} \otimes \frac{\partial \boldsymbol{\Phi}_{Q-k+1}}{\partial \boldsymbol{\Phi}_i}) \\ &= -\frac{1}{2}\left[\frac{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'}{\sigma^2} + (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{ii} - (\boldsymbol{I}_{m_i} \otimes \boldsymbol{\Phi}_i)^{-1}\right] \\ &\quad * (\boldsymbol{I}_{m_i} \otimes \text{vec}(\boldsymbol{I}_{q_i})\,\text{vec}'(\boldsymbol{I}_{q_i})) \\ &= -\frac{1}{2}\sum_{k=1}^{m_i}\left[\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2} + (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{ii(k)} - \boldsymbol{\Phi}_i^{-1}\right] \\ &\quad * \text{vec}(\boldsymbol{I}_{q_i})\,\text{vec}'(\boldsymbol{I}_{q_i}) \\ &= -\frac{1}{2}\sum_{k=1}^{m_i}\left[\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2} + (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_A)^{ii(k)} - \boldsymbol{\Phi}_i^{-1}\right]\end{aligned}$$

We can now write the gradient of the profiled log-likelihood as

$$\sum_{i=1}^{Q} \frac{\partial \ell}{\partial\{\Phi_i\}} * \frac{d\Phi_i}{d\theta}$$

$$= -\frac{1}{2} \sum_{i=1}^{Q} \sum_{k,l} \left\{ \sum_{j=1}^{m_i} \left[\frac{\widehat{b_{i(j)}}_{ML}(\theta) \widehat{b_{i(j)}}_{ML}(\theta)'}{\widehat{\sigma^2}_{ML}(\theta)} + (Z'Z + \Phi_A)^{ii(j)} - \Phi_i^{-1} \right] \right\}_{kl} \frac{d\{\Phi_i\}_{kl}}{d\theta} \quad \text{(B-17)}$$

2.1 Gradient of the Log-Restricted-Likelihood

The partial derivative of (B-13) with respect to Φ_A is given by

$$\frac{\partial}{\partial\{\Phi_A\}} \ell_R(\sigma^2, \Phi_A | y)$$

$$= -\frac{1}{2} \left[\frac{\partial \log |Z'Z + \Phi_A|}{\partial\{\Phi_A\}} - \frac{\partial \log |\Phi_A|}{\partial\{\Phi_A\}} \right]$$

$$- \frac{1}{2\sigma^2} \frac{\partial}{\partial\{\Phi_A\}} (y - X\beta)' (I - Z(Z'Z + \Phi_A)^{-1} Z')(y - X\beta)$$

$$- \frac{1}{2} \frac{\partial \log \left| X' \left(I - Z(Z'Z + \Phi_A)^{-1} Z' \right) X \right|}{\partial\{\Phi_A\}}$$

$$= -\frac{1}{2} \left[(Z'Z + \Phi_A)^{-1} - \Phi_A^{-1} \right]$$

$$- \frac{1}{2\sigma^2} (Z'Z + \Phi_A)^{-1} Z'X (y - X\beta)$$

$$(y - X\beta)' X' Z (Z'Z + \Phi_A)^{-1}$$

$$- \frac{1}{2} (Z'Z + \Phi_A)^{-1} Z'X \Sigma_X X'Z (Z'Z + \Phi_A)^{-1}$$

$$= -\frac{1}{2} \left[\frac{\widehat{b}\widehat{b}'}{\sigma^2} + (Z'Z + \Phi_A)^{-1} - \Phi_A^{-1} \right.$$

$$\left. + (Z'Z + \Phi_A)^{-1} Z'X \Sigma_X X'Z (Z'Z + \Phi_A)^{-1} \right]$$

Let

$$\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}+\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}\boldsymbol{Z}'\boldsymbol{X}\boldsymbol{\Sigma}_{\boldsymbol{X}}\boldsymbol{X}'\boldsymbol{Z}\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}$$
$$=\begin{bmatrix}\boldsymbol{S}_{11} & \boldsymbol{S}_{12} & \dots & \boldsymbol{S}_{1Q}\\ \boldsymbol{S}_{21} & \boldsymbol{S}_{22} & \dots & \boldsymbol{S}_{2Q}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{S}_{Q1} & \boldsymbol{S}_{Q2} & \dots & \boldsymbol{S}_{QQ}\end{bmatrix}$$

where $\boldsymbol{S}_{ij}$ is of size $m_iq_i \times m_jq_j$ and

$$\boldsymbol{S}_{ii}=\begin{bmatrix}\boldsymbol{S}_{ii(1)} & \boldsymbol{S}_{ii(12)} & \dots & \boldsymbol{S}_{ii(1m_i)}\\ \boldsymbol{S}_{ii(21)} & \boldsymbol{S}_{ii(2)} & \dots & \boldsymbol{S}_{ii(2m_i)}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{S}_{ii(m_i1)} & \boldsymbol{S}_{ii(m_i2)} & \dots & \boldsymbol{S}_{ii(m_i)}\end{bmatrix}$$

Using (B-16),

$$\begin{aligned}\frac{\partial \ell_R}{\partial\{\boldsymbol{\Phi}_i\}} &= \frac{\partial \ell_R}{\partial\{\boldsymbol{\Phi}_A\}} * \frac{\partial D_A}{\partial\{\boldsymbol{\Phi}_i\}}\\ &= -\frac{1}{2}\Big[\frac{\widehat{\boldsymbol{b}}\widehat{\boldsymbol{b}}'}{\sigma^2}+\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}\\ &\qquad +\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}\boldsymbol{Z}'\boldsymbol{X}\boldsymbol{\Sigma}_{\boldsymbol{X}}\boldsymbol{X}'\boldsymbol{Z}\left(\boldsymbol{Z}'\boldsymbol{Z}+\boldsymbol{\Phi}_A\right)^{-1}\\ &\qquad -\boldsymbol{\Phi}_A^{-1}\Big] * \bigoplus_{k=1}^{Q}(\boldsymbol{I}_{m_{Q-k+1}}\otimes\frac{\partial\boldsymbol{\Phi}_{Q-k+1}}{\partial\{\boldsymbol{\Phi}_i\}})\\ &= -\frac{1}{2}\left[\frac{\widehat{\boldsymbol{b}}_i\widehat{\boldsymbol{b}}_i'}{\sigma^2}+\boldsymbol{S}_{ii}-(I_{m_i}\otimes\boldsymbol{\Phi}_i)^{-1}\right] * (\boldsymbol{I}_{m_i}\otimes\operatorname{vec}(\boldsymbol{I}_{q_i})\operatorname{vec}'(\boldsymbol{I}_{q_i}))\\ &= -\frac{1}{2}\sum_{k=1}^{m_i}\left[\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2}+\boldsymbol{S}_{ii(k)}-\boldsymbol{\Phi}_i^{-1}\right] * \operatorname{vec}(\boldsymbol{I}_{q_i})\operatorname{vec}'(\boldsymbol{I}_{q_i})\\ &= -\frac{1}{2}\sum_{k=1}^{m_i}\left[\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2}+\boldsymbol{S}_{ii(k)}-\boldsymbol{\Phi}_i^{-1}\right]\end{aligned}$$

2.2 Efficient calculation of the gradients

As discussed in §1.2 we do not use the analytical formulas derived here directly when computing the gradients numerically. Instead we use the decomposed matrices.

First of all, for $i > j$ define

$$\boldsymbol{T}_{i,j} = \begin{bmatrix} \boldsymbol{R}_{ii(1)}^{-1}\boldsymbol{R}_{ij(11)}\boldsymbol{R}_{jj(1)}^{-1} \\ \vdots \\ \boldsymbol{R}_{ii(1)}^{-1}\boldsymbol{R}_{ij(m_{ij(1)}1)}\boldsymbol{R}_{jj(1)}^{-1} \\ \vdots \\ \boldsymbol{R}_{ii(1)}^{-1}\boldsymbol{R}_{ij(1m_j)}\boldsymbol{R}_{jj(m_j)}^{-1} \\ \vdots \\ \boldsymbol{R}_{ii(1)}^{-1}\boldsymbol{R}_{ij(m_{ij(m_j)}m_j)}\boldsymbol{R}_{jj(m_j)}^{-1} \end{bmatrix}$$

Generalizing the approach taken in DebRoy and Bates (2003) we compute the QR decompostion

$$\begin{bmatrix} \widehat{\boldsymbol{b}_{i(1)}}'_{ML}/\widehat{\sigma}_{ML} \\ \boldsymbol{R}_{ii(1)}^{-1} \\ \vdots \\ \widehat{\boldsymbol{b}_{i(m_i)}}'_{ML}/\widehat{\sigma}_{ML} \\ \boldsymbol{R}_{ii(m_i)}^{-1} \\ \boldsymbol{T}_{i,i+1} \\ \vdots \\ \boldsymbol{T}_{i,Q} \end{bmatrix} = U_i A_i \tag{B-18}$$

Then the gradient of the profiled log-likelihood is given by

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = -\frac{1}{2}\sum_{i=1}^{Q}\sum_{j=1}^{q_i}\sum_{k=1}^{q_k}\left((\boldsymbol{A}_i'\boldsymbol{A}_i)_{jk} - m\boldsymbol{\Phi}_i^{jk}\right)\frac{d}{d\boldsymbol{\theta}}\{\boldsymbol{\Phi}_i\}_{jk}(\boldsymbol{\theta}) \tag{B-19}$$

In the case of the REML gradient, there are some extra terms in the

decomposition

$$\begin{bmatrix} \widehat{\boldsymbol{b}_{i(1)}}'_{ML}/\widehat{\sigma}_{ML} \\ \boldsymbol{R}^{-1}_{ii(1)} \\ \left(\boldsymbol{R}^{-1}_{ii(1)}\boldsymbol{R}_{i0(1)}\boldsymbol{R}^{-1}_{00}\right)' \\ \vdots \\ \widehat{\boldsymbol{b}_{i(m_i)}}'_{ML}/\widehat{\sigma}_{ML} \\ \boldsymbol{R}^{-1}_{ii(m_i)} \\ \left(\boldsymbol{R}^{-1}_{ii(m_i)}\boldsymbol{R}_{i0(m)}\boldsymbol{R}^{-1}_{00}\right)' \\ \boldsymbol{T}_{i,i+1} \\ \vdots \\ \boldsymbol{T}_{i,Q} \end{bmatrix} = U_{Ri}A_{Ri} \tag{B-20}$$

Then the gradient of the profiled log-restricted-likelihood is given by

$$\frac{d\tilde{\ell}}{d\boldsymbol{\theta}} = -\frac{1}{2}\sum_{i=1}^{Q}\sum_{j=1}^{q_i}\sum_{k=1}^{q_k}\left((\boldsymbol{A}'_{Ri}\boldsymbol{A}_{Ri})_{jk} - m\boldsymbol{\Phi}_i^{jk}\right)\frac{d}{d\boldsymbol{\theta}}\{\boldsymbol{\Phi}_i\}_{jk}(\boldsymbol{\theta}) \tag{B-21}$$

3 ECME Algorithm for linear mixed effects models

DebRoy and Bates (2003) presented an ECME algorithm for single-level LME models. Here we extend that algorithm to multiple nested levels of random effects.

Using $\boldsymbol{b}$ as the missing data, the log-likelihood of the complete data is

$$\begin{aligned}\ell\left(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{b}\right) = -\frac{1}{2}\Big[&(n + \sum_{k=1}^{Q} m_k q_k)\log(2\pi\sigma^2) \\ &+ \frac{1}{\sigma^2}\left\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}\right\|^2 - \log|\boldsymbol{\Phi}_A| + \frac{1}{\sigma^2}\boldsymbol{b}'\boldsymbol{\Phi}_A\boldsymbol{b}\Big]\end{aligned} \tag{B-22}$$

In subsequent discussion, we will assume that the current values of the parameters are $\boldsymbol{\beta}_0$, $\boldsymbol{\sigma}_0^2$ and $\boldsymbol{\theta}_0$. These can be either the starting values or the values obtained from the last E and M steps. The parameter estimates to be obtained at the end of the current E and M steps are $\boldsymbol{\beta}_1$, $\boldsymbol{\sigma}_1^2$ and $\boldsymbol{\theta}_1$. We also use $\boldsymbol{\Phi}_{A0} = \boldsymbol{\Phi}_A(\boldsymbol{\theta}_0)$ and $\boldsymbol{\Phi}_{A1} = \boldsymbol{\Phi}_A(\boldsymbol{\theta}_1)$.

The conditional distribution of the random effects is given by

$$\boldsymbol{b}|\boldsymbol{y}, \beta_0, \boldsymbol{\sigma}_0^2, \boldsymbol{\theta}_0 \sim \mathcal{N}\left(\widehat{\boldsymbol{b}}, (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\boldsymbol{\sigma}_0^2\right) \tag{B-23}$$

where $\widehat{\boldsymbol{b}}$ is the expected value of $\boldsymbol{b}$ and is given by

$$\widehat{\boldsymbol{b}} = (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\boldsymbol{Z}'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}_0) \tag{B-24}$$

When evaluated at $\widehat{\boldsymbol{\Phi}_A}$ these are called the *best linear unbiased predictors* (BLUPs) of $\boldsymbol{b}$.

3.1 The E step

In this step we compute $Q(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{\beta}_0, \sigma_0^2, \boldsymbol{\theta}_0)$, the conditional expectation of $\ell\left(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{b}\right)$ as defined in equation (B-22).

$$\begin{aligned}
&Q(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{\beta}_0, \sigma_0^2, \boldsymbol{\theta}_0) \\
&\quad = \mathrm{E}_{\boldsymbol{b}|\boldsymbol{y}}\left[-\frac{n + \sum_{k=1}^{Q} m_k q_k}{2}\log(2\pi\sigma^2) - \frac{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}\|^2}{2\sigma^2}\right. \\
&\qquad\qquad \left. + \frac{\log|\boldsymbol{\Phi}_A|}{2} - \frac{\|\boldsymbol{\Phi}_A^{1/2}\boldsymbol{b}\|^2}{2\sigma^2}\right] \\
&\quad = -\frac{n + \sum_{k=1}^{Q} m_k q_k}{2}\log(2\pi\sigma^2) - \mathrm{E}_{\boldsymbol{b}|\boldsymbol{y}}\frac{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}\|^2}{2\sigma^2} \\
&\qquad + \frac{1}{2}\log|\boldsymbol{\Phi}_A| - \frac{\mathrm{E}_{\boldsymbol{b}|\boldsymbol{y}}\|\boldsymbol{\Phi}_A^{1/2}\boldsymbol{b}\|^2}{2\sigma^2} \\
&\quad = -\frac{n + \sum_{k=1}^{Q} m_k q_k}{2}\log(2\pi\sigma^2) + \frac{1}{2}\log|\boldsymbol{\Phi}_A| \\
&\qquad - \left\{\frac{\|\boldsymbol{\Phi}_A^{1/2}\widehat{\boldsymbol{b}}\|^2}{2\sigma^2} + \frac{\mathrm{tr}\left[\boldsymbol{\Phi}_A\sigma_0^2(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\right]}{2\sigma^2}\right\} \\
&\qquad - \left\{\frac{\|\boldsymbol{y} - \boldsymbol{Z}\widehat{\boldsymbol{b}} - \boldsymbol{X}\boldsymbol{\beta}\|^2}{2\sigma^2} + \frac{\mathrm{tr}\left[\boldsymbol{Z}'\boldsymbol{Z}\sigma_0^2(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\right]}{2\sigma^2}\right\}
\end{aligned} \tag{B-25}$$

3.2 The M step

We use the following ECME scheme

1. Maximize Q over $\boldsymbol{\theta}$ by fixing σ^2 at $\widehat{\sigma}_0^2$ and $\boldsymbol{\beta}$ at $\boldsymbol{\beta}_0$ to obtain $\boldsymbol{\theta}_1$.
2. Maximize ℓ over $\boldsymbol{\beta}$ and σ^2 by fixing $\boldsymbol{\theta}$ at $\boldsymbol{\theta}_1$ to obtain $\boldsymbol{\beta}_1$ and $\widehat{\sigma}_1^2$.

We only need to maximize Q with respect to $\boldsymbol{\theta}$, which is equivalent to maximizing with respect to all the $\boldsymbol{\Phi}_i, i = 1, \ldots, Q$. First we must compute the partial derivative of (B-25) with respect to $\boldsymbol{\Phi}_A$.

$$\begin{aligned}
&\frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}} Q(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\Phi}_A | \boldsymbol{y}, \boldsymbol{\beta}_0, \sigma_0^2, \boldsymbol{\Phi}_0) \\
&\quad = -\frac{1}{2\sigma^2} \frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}} \left[\|\boldsymbol{\Phi}_A^{1/2} \widehat{\boldsymbol{b}}\|^2 + \operatorname{tr}\left[\boldsymbol{\Phi}_A \sigma_0^2 (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1} \right] \right] \\
&\qquad + \frac{1}{2} \frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}} \log |\boldsymbol{\Phi}_A| \qquad \text{(B-26)} \\
&\quad = -\frac{1}{2} \left[\frac{\widehat{\boldsymbol{b}}\widehat{\boldsymbol{b}}'}{\sigma^2} + \frac{\sigma_0^2}{\sigma^2} (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1} - \boldsymbol{\Phi}_A^{-1} \right]
\end{aligned}$$

Using (B-16),

$$\begin{aligned}
&\frac{\partial}{\partial\{\boldsymbol{\Phi}_i\}} Q(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\Phi}_A | \boldsymbol{y}, \boldsymbol{\beta}_0, \sigma_0^2, \boldsymbol{\Phi}_0) \\
&\qquad = \frac{\partial Q}{\partial\{\boldsymbol{\Phi}_A\}} * \frac{\partial \boldsymbol{\Phi}_A}{\partial\{\boldsymbol{\Phi}_i\}} \\
&\qquad = -\frac{1}{2} \left[\frac{\widehat{\boldsymbol{b}}\widehat{\boldsymbol{b}}'}{\sigma^2} + \frac{\sigma_0^2}{\sigma^2} (\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1} - \boldsymbol{\Phi}_A^{-1} \right] \\
&\qquad\quad * \bigoplus_{k=1}^{Q} (\boldsymbol{I}_{m_{Q-k+1}} \otimes \frac{\partial \boldsymbol{\Phi}_{Q-k+1}}{\partial\{\boldsymbol{\Phi}_i\}}) \\
&\qquad = -\frac{1}{2} \left[\frac{\widehat{\boldsymbol{b}}_i \widehat{\boldsymbol{b}}_i'}{\sigma^2} + \frac{\sigma_0^2}{\sigma^2} \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0} \right)^{ii} - (I_{m_i} \otimes \boldsymbol{\Phi}_i)^{-1} \right] \\
&\qquad\quad * (\boldsymbol{I}_{m_i} \otimes \operatorname{vec}(\boldsymbol{I}_{q_i}) \operatorname{vec}'(\boldsymbol{I}_{q_i})) \\
&\qquad = -\frac{1}{2} \sum_{k=1}^{m_i} \left[\frac{\widehat{\boldsymbol{b}}_{i(k)} \widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2} + \frac{\sigma_0^2}{\sigma^2} \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0} \right)^{ii(k)} - \boldsymbol{\Phi}_i^{-1} \right] \\
&\qquad\quad * \operatorname{vec}(\boldsymbol{I}_{q_i}) \operatorname{vec}'(\boldsymbol{I}_{q_i})) \\
&\qquad = -\frac{1}{2} \sum_{k=1}^{m_i} \left[\frac{\widehat{\boldsymbol{b}}_{i(k)} \widehat{\boldsymbol{b}}_{i(k)}'}{\sigma^2} + \frac{\sigma_0^2}{\sigma^2} \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0} \right)^{ii(k)} - \boldsymbol{\Phi}_i^{-1} \right]
\end{aligned}$$

Equating to the zero matrix and substituting $\sigma^2 = \sigma_0^2 = \widehat{\sigma^2}_{ML}$

$$\boldsymbol{\Phi}_{i1}^{-1} = \frac{1}{m_i}\sum_{i=1}^{m_i}\left\{\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}'_{i(k)}}{\widehat{\sigma^2}_{ML}} + \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0}\right)^{ii(k)}\right\}$$

$$\Rightarrow \quad \boldsymbol{\Phi}_{i1} = m_i\left[\sum_{i=1}^{m_i}\left\{\frac{\widehat{\boldsymbol{b}}_{i(k)}\widehat{\boldsymbol{b}}'_{i(k)}}{\widehat{\sigma^2}_{ML}} + \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0}\right)^{ii(k)}\right\}\right]^{-1} \qquad \text{(B-27)}$$

We can use the decomposition (B-18) to compute the update more efficiently as $\boldsymbol{\Phi}_{i1}^{1/2} = \sqrt{m_i}\left(\boldsymbol{A}_i^{-1}\right)'$.

3.3 ECME algorithm for REML

Again we generalize the ECME algorithm in DebRoy and Bates (2003) for REML estimation. As explained there, we extend the complete data set to include $\boldsymbol{\beta}$ as if it were another level of random effects but with a precision matrix $\boldsymbol{\Phi}_0 = \boldsymbol{0}$. (In a Bayesian formulation this would correspond to an improper, "locally uniform" prior distribution.) We note that the complete data likelihood $\ell_R\left(\sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{b}\right)$ has exactly the same expression as in (B-22). In the E step we must take expectation of (B-25) with respect to $\boldsymbol{\beta}$ by regarding $\boldsymbol{\beta}_1$ as $\boldsymbol{\beta}$ using the conditional distribution of $\boldsymbol{\beta}$

$$\boldsymbol{\beta}|\boldsymbol{y}, \sigma_0^2, \boldsymbol{\theta} \sim \mathcal{N}\left(\widehat{\boldsymbol{\beta}}_{ML}, \left(\boldsymbol{X}'\boldsymbol{\Sigma}_{X0}^{-1}\boldsymbol{X}\right)^{-1}\sigma_0^2\right)$$

where $\boldsymbol{\Sigma}_{X0}$ is defined by using $\boldsymbol{\Phi}_A = \boldsymbol{\Phi}_{A0}$ in $\boldsymbol{\Sigma}_X$. This gives us $Q_R = E_{\boldsymbol{\beta},\boldsymbol{b}|\boldsymbol{y}}\ell_R\left(\sigma^2, \boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{b}\right)$.

As we noted in DebRoy and Bates (2003), we only need to deal with the terms which involve both $\boldsymbol{\Phi}_A$ and $\boldsymbol{\beta}$ in (B-25) for maximizing with respect to $\boldsymbol{\Phi}_A$. The only additional term in the derivative is

$$-\frac{\partial}{\partial\{\boldsymbol{\Phi}_A\}}\frac{\sigma_0^2}{2\sigma^2}\operatorname{tr}\left[\boldsymbol{\Phi}_A(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\boldsymbol{Z}'\boldsymbol{X}\boldsymbol{\Sigma}_{X0}\boldsymbol{X}'\boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\right]$$
$$= -(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}\boldsymbol{Z}'\boldsymbol{X}\boldsymbol{\Sigma}_{X0}\boldsymbol{X}'\boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{\Phi}_{A0})^{-1}$$

We must take this term into account when computing the EM update. That is done in the decompostion (B-20) so that we can calculate the update using $\boldsymbol{\Phi}_{i1}^{1/2} = \sqrt{m_i}\left(\boldsymbol{A}_{Ri}^{-1}\right)'$.

References

Saikat DebRoy and Douglas M. Bates. Computational methods for single level linear mixed-effects models. Technical Report 1073, Department of Statistics, University of Wisconsin-Madison, 2003.

Harvey Goldstein. *Multilevel models in education and social research.* Oxford University Press, 1987.

Nan M. Laird and James H. Ware. Random-effects models for longitudinal data. *Biometrics*, 38:963–974, 1982.

José C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-PLUS.* Statistics and Computing. Springer-Verlag Inc, 2000. ISBN 0-387-98957-9.

Stephen W. Raudenbush and Anthony S. Bryk. *Hierarchical Linear Models: Applications and Data Analysis Methods.* Sage, second edition, 2002. ISBN 0-7619-1904-X.

Gerald S. Rogers. *Matrix Derivatives.* Marcell Dekker, Inc., New York and Basel, 1980.

Appendix C: Distributed Statistical Computing

Conference Paper

Converting a large R package to S4 classes and methods

Douglas M. Bates and Saikat DebRoy*
Department of Statistics
University of Wisconsin – Madison

February 19, 2003

Abstract

The `nlme` package for fitting and examining linear and nonlinear mixed-effects models in R is a required package and also one of the largest R packages. In the first phase of a project to extend the capabilities of the nlme package to include generalized linear mixed models (glmm's), we reimplemented linear mixed-effects (lme) models using 'S4' classes and methods, as described in John Chambers' book "Programming with Data" and as implemented in the `methods` package for R. Our general goals for this phase are to incorporate new theoretical and computational developments for the lme model and to provide a faster, cleaner implementation of lme fits in R while including hooks for later extensions to the glmm model and the nlme model. In particular, we use our reStruct (random-effects structure) class in iterative PQL fits for glmm's, based on Brian Ripley's function `glmmPQL` from the `MASS` package.

As described in "Programming with Data", classes, slots and inheritance relationships must be declared explicitly when using the methods package. Although such formal declarations require package authors to be more disciplined than when using informal 'S3' classes, they provide assurance that each object in a class has the required slots and that the names and classes of data in the slots are consistent. This is important to us because we are trying to achieve both efficiency and flexibility. We provide flexibility by defining many classes and methods and by using multiple-argument signatures in method declarations. We achieve efficiency by implementing many methods in C code using the `.Call` interface and through liberal use of `GET_SLOT` and `SET_SLOT` within the C code.

*This work is supported by U.S. Army Medical Research and Materiel Command under Contract No. DAMD17-02-C-0119. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

We feel that the new implementation is much cleaner and easier to understand than the previous implementation, due in large part to the more extensive use of classes and methods. It is definitely faster and can handle larger problems than the previous implementation.

1 Introduction

The `nlme` package (Pinheiro and Bates, 2000) for fitting and examining linear and nonlinear mixed-effects models is a large R package. It consists of more than 500 R functions, 3500 lines of C code, and 40 data sets plus documentation and examples. The reason that there are so many functions in a package that is devoted to fitting just one general type of statistical model is to provide flexibility in specifying and examining the models. We also want to fit mixed-effects models efficiently.

To organize the large number functions applied to different types of objects we created many classes of objects representing, for example, grouped data (groupedData), linear mixed-effects model structures (lmeStruct), nonlinear mixed-effects model structures, random-effects structures (reStruct), positive-definite parameterized matrices (pdMat), correlation structures (corStruct), variance functions (varFunc) and many different kinds of fitted models or summaries or plots derived from fitted models. Most of the 500 functions are methods for different classes of objects.

The computational methods described in Pinheiro and Bates (2000, ch. 2,7) for efficiently evalating and profiling the log-likelihood or log-restricted-likelihood of a linear or nonlinear mixed-effects model with multiple, nested levels of random effects are quite formidable. Model matrices corresponding to the fixed-effects or to the random-effects terms in the statistical model are combined and decomposed repeatedly during the iterative optimization of the objective function to determine the parameter estimates. To achieve a reasonable level of efficiency in fitting models we coded the compute-intensive parts of the calculations in C.

We have begun a project to extend the capabilities of `nlme` to fit generalized linear mixed models (Raudenbush and Bryk, 2002, ch. 10), beginning with the method implemented by Brian Ripley in the `glmmPQL` function from the `MASS` package but also implementing estimation methods based on Laplacian and adaptive Gauss-Hermite approximations to the integral of the conditional density of the random effects.

In the first phase of this project we have reimplemented the data structures and computational algorithms for linear mixed models as 'S4' classes and methods. Our objectives for this reimplementation are:

- To encapsulate the underlying structures for linear mixed models in such a way that they can be extended to generalized linear mixed models and to nonlinear mixed models.

- To incorporate new theoretical and computational developments for the

lme model. We have derived the analytic gradient of the profiled log-likelihood (or log-restricted likelihood) of a linear mixed model (DebRoy and Bates, 2003a,b) and have related the gradient results to an ECME (expectation conditional maximization either) optimization step. The analytic gradient allows for faster and, more importantly, more stable optimization.

- To convert the numerical linear algebra calls from Linpack and BLAS-1 calls to Lapack (Anderson et al., 1992) and BLAS levels 1, 2, and 3. Lapack provides state-of-the-art algorithms and can provide a substantial performance boost when ATLAS (Automatically Tuned Linear Algebra Software) implementations of the BLAS and some Lapack routines are available.

- To switch all calls of C code to the `.Call` interface so that entire R objects can be passed to and from the C code. This also allows direct access to the slots of S4 classed objects from within C code.

- To monitor the number of copies of objects that are created, especially those created within iterative algorithms. In §5 we discuss an example of a model fit to 375,000 observations on 135,000 subjects grouped into 3722 groups. The model matrices for the fixed effects can have as many as 40 or 50 columns, or about 150 MB for each copy of the model matrix and information derived from it. We need at least three arrays of this size to keep track of all the information we use. We do not want to create more than that if we can avoid doing so.

1.1 S3 versus S4 classes and methods

Object-oriented programming is a powerful tool for organizing the representation of information (classes) and the actions that are applied to these representations (methods). A system of classes and methods for the S language was introduced in Chambers and Hastie (1992). We will call this the 'S3' class system, to distinguish it from the 'S4' class system described in Chambers (1998) and implemented for R in the `methods` package. Unlike object-oriented languages such as Java and C++ where methods are associated with a class definition, both the S3 and the S4 systems associate methods with the combination of a generic function and the classes of one or more of the arguments to that function.

S3 classes are informal: the class of an object is determined by its class attribute, which should consist of one or more character strings, and methods are found by combining the name of the generic function with the class of the first argument to the function. If a function having this combined name is on the search path, it is assumed to be the appropriate method. Classes and their contents are not formally defined in the S3 system - at best there is a "gentleman's agreement" that objects in a class will have certain structure with certain component names.

By contrast, S4 classes must be defined explicitly. The number of slots in objects of the class, and the names and classes of the slots, are established at the time of class definition. During computation with objects from the class they can be validated against the definition. As in many other object-oriented systems, an S4 class can be declared to inherit from another class so S4 classes can be arranged in a hierarchy.

S4 also requires formal declarations of methods, unlike the informal system of using function names to identify a method in S3. An S4 method is declared by a call to `setMethod` giving the name of the generic and the "signature" of the arguments. The signature identifies the classes or one or more named arguments to the generic function. Special meta-classes named `ANY` and `missing` can be used in the signature.

S4 generic functions can be declared by a call to `setGeneric` or they can be automatically created by declaring a method for an existing function, in which case the function becomes generic and the current definition becomes the default method.

2 Package conversion: creating S4 classes

The principle generic functions for mixed-effects models and the methods associated with them were already defined in version 3.1 of the nlme package. Although these generics and methods would be modified to some extent during the conversion to S4 classes and methods, we could initially assume these would stay as they are and concentrate instead on determining what classes should be defined and how we should define them.

We could use the informal set of classes from the S3 version as a guide when formulating the S4 classes. We found, however, that we frequently reconsidered the structure of the classes during the conversion and usually ended up adding more slots to the classes than had been present in the informal, 'S3' version.

Consider, for example, the `pdMat` class of parameterized, positive definite, symmetric matrices. It is a virtual class (which means that there will be objects of classes that inherit from `pdMat` but there will never be objects whose only class is `pdMat`). The matrices represented by objects that inherit from this class are determined by a non-redundant, unconstrained vector of parameters. In some parameterizations the dimensions of the matrix can be determined from the length of the parameter vector but in others, such as `pdIdent`, representing multiples of the identity, or `pdCompSymm`, representing matrices with compound symmetry, the parameter vector has a fixed length and the number of columns in the matrix must be stored separately. We decided to add an `Ncol` slot to all the `pdMat` classes and did so by declaring it in the virtual `pdMat` class. In fact, the `pdMat` class declares six slots that must be present in all classes inheriting from it.

```
setClass("pdMat",       # parameterized positive-definite matrices
  representation(form="formula",     # a model-matrix formula
                 Names="character", # column (and row) names
```

```
                param="numeric",   # parameter vector
                Ncol="integer",    # number of columns
                factor="matrix",   # factor of the pos-def matrix
                logDet="numeric"   # logarithm of the absolute value
                ## of the determinant of the factor (i.e. half
                ## the logarithm of the determinant of the matrix)
                ),
  prototype(form=formula(NULL),
            Names=character(0),
            param=numeric(0),
            Ncol=as.integer(0),
            factor=matrix(numeric(0),0,0),
            logDet=numeric(0))
  )
```

After this definition most of the class definitions for other pdMat classes are trivial.

```
setClass("pdSymm", "pdMat")        # general symmetric pd matrices
setClass("pdScalar", "pdSymm")     # special case of positive scalars
setClass("pdLogChol", "pdSymm")    # default parameterization
setClass("pdNatural", "pdSymm")    # log sd and Fisher's z of correlation
setClass("pdMatrixLog", "pdSymm") # matrix logarithm parameterization
setClass("pdDiag", "pdMat")        # diagonal pd matrices
setClass("pdIdent", "pdMat")       # positive multiple of the identity
setClass("pdCompSymm", "pdMat")    # compound symmetric pd matrices
```

Increasing the number of slots may be an inevitable consequence of revising the package (we tend to add capabilities more frequently than we remove them) but it may also be related to the fact that S4 classes must be declared explicity and hence we consider the components or slots of the classes and the relationships between the classes more carefully.

3 Calling C functions with .Call

The `.Call` interface, through which a programmer can pass raw R objects to C code and receive raw R objects from the C code, has been part of R for several years. It was inspired by the `.Call` interface for S described in Chambers (1998). Several C macros for working with S4 classed objects, including `GET_SLOT`, `SET_SLOT`, `MAKE_CLASS` and `NEW`, all described in Chambers (1998) are are now available in R, or will be in R-1.7.0. (Note that the macro `NEW_OBJECT` is preferred to `NEW` when writing code for R only. These two macros have the same effect but `NEW_OBJECT` is less likely to conflict with other definitions.)

The combination of the formal classes of S4, the `.Call` interface, and these macros allows a programmer to manipulate S4 classed objects in C code nearly as easily as in R code. A common idiom is to have an S4 method call C code

through the .Call interface. In the C code the values of slots are extracted with `GET_SLOT` and either modified in place or used to create slots for new objects. Such new objects are created and populated by calls to `MAKE_CLASS`, `NEW_OBJECT`, and `SET_SLOT`.

Because the C code is called from a method, the programmer can be confident of the classes of the objects passed in the call and the classes of the slots of those objects. Much of the checking of classes or modes and possible coersion of modes that is common in C code called from R can be bypassed.

We found that we would initially write methods in R then translate them into C if warranted. The nature of our calculations, frequently involving multiple decompositions and manipulations of sections of arrays, was such that the calculations could be expressed in R but not very cleanly. Once we had the R version working satisfactorily we could translate into C the parts that were critical for performance or were awkward to write in R. An important advantage of this mode of development is that we could use the same slots in the C version as in the R version and create the same types of objects to be returned.

We feel that defining S4 classes and methods in R then translating parts of method definitions to C functions called through `.Call` is an extremely effective mode for numerical computation. Programmers who have experience working in C++ or Java may initially find it more convenient to define classes and methods in the compiled language and perhaps define a parallel series of classes in R. (We did exactly that when creating the Matrix package for R.) We encourage such programmers to try instead this method of defining only one set of classes, the S4 classes in R, and use these classes in both the interpreted language and the compiled language.

3.1 Using replacement methods

The `.Call` interface is a powerful tool and, like many powerful tools, must be used carefully if you are to avoid hurting yourself with it. The programmer must be aware that the arguments are not copied when they are passed through `.Call` even though the semantics of R function calls require that arguments must not be modified by a function call. If you are to modify the value of an R object passed as an argument you must somehow copy its storage, usually by duplicating it or coercing it to another mode, before making any changes. Failure to do so can result in bugs that are extremely difficult to diagnose.

Duplicating or coercing R objects will usually require that the result be protected from the garbage collector by a call to the `PROTECT` macro. The effect of all calls to `PROTECT` must be undone by calling `UNPROTECT` before returning from the C function. Keeping track of what has been protected can sometimes be tedious.

There is a one exception to the "don't modify the arguments" rule: a replacement function or a replacement method is allowed to modify its first argument. Because the class of the result is the same as the class of the first argument it is common to use this argument as the return value, after suitable modification of its contents. We found that we used replacement methods more frequently

in our code than we had first expected. We tended to think in steps of creating an object then modifying it according to the values of other objects.

For example, a linear mixed-effects model is represented by an `reStruct` object. The core part of the code to fit such a model is

```
re <- reStruct(fixed = fixed, random = random,
               data = eval(mCall, parent.frame()),
               REML = method != "ML")
EMsteps(re) <- controlvals
LMEoptimize(re) <- controlvals
```

where we construct the `reStruct` object, perform some number of EM updates on it then perform general nonlinear optimization on it.

4 Use of Lapack and ATLAS

Linpack and Eispack routines are used for numerical linear algebra in R since its inception and are part of the R API. The Linpack and Eispack packages have been largely supercede by Lapack (Anderson et al., 1992) which provides, in some cases, better algorithms and, in nearly all cases, more effective implementations of the algorithms. Some of the effectiveness of the implementations, especially for large arrays, comes from more extensive use of the Basic Linear Algebra Subroutines (BLAS). As the name implies, these are basic routines for doing operations like multiplying two matrices or replacing $\boldsymbol{y}$ by $a\boldsymbol{x} + \boldsymbol{y}$. Even in sophisticated linear algebra algorithms, the majority of the numerical computation takes place in these basic operations, hence it is worthwhile devoting considerable effort to optimizing these routines. ATLAS (Automatically Tuned Linear Algebra Software) is a collection of highly optimized BLAS routines that can be compiled for different architectures. The combination of Lapack and ATLAS can give a considerable performance boost to algorithms that use numerical linear algebra extensively.

The R Core Development Group (primarily Brian Ripley) has been migrating R from Linpack and Eispack to Lapack. Beginning with R-1.7.0 the double precision Lapack routines and some of the double precision complex Lapack routines will be part of the R API. We converted all the linear algebra in the lme calculations to Lapack, with gratifying results as described in the next section.

5 Timing results

Rodriguez and Goldman (1995) simulated 100 sets of 2445 binary responses grouped into 1558 families in 161 communities and fit generalized linear mixed models with two levels of random effects to these. The implementation of `glmmPQL` in the new version of the package is roughly 5 times as fast on these fits as the previous version that used repeated calls to `lme`.

We also fit a linear mixed-effects model to 378047 mathematics scores of 134713 students on the Texas Assessment of Academic Skills (TAAS). The data were all the test scores of students in grades 3 to 8 in Dallas, Texas during 1994 to 2000. The particular model that we fit had a fixed-effects vector of length 47, resulting in very large model matrices. A fit with the previous version of the lme function took 993 seconds of user time (1093 seconds elapsed time) on a 2.0 GHz Pentium-4 machine with 1.0 GB of PC-2700 memory running Debian GNU/Linux. The new version of lme took 221 seconds user time (345 seconds elapsed time) on the same machine.

6 Conclusions

We feel that we have met our objectives in reimplementing the lme part of the `nlme` package using 'S4' classes and methods. Although the performance boost from using Lapack and ATLAS is gratifying, we feel that the biggest gain is in making the code much cleaner and easier to understand and in exposing interfaces that can be used by models that extend the linear mixed-effects model.

Code clarity is enhanced by the fact that S4 classes and the .Call interface allow programmers to work with the same class definitions in R code and in C code. We have also found that liberal use of replacement functions and methods allows us to maintain control of the number of copies of objects being created.

References

E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammaring, A. McKenney, S. Ostrouchov, and D. Sorensen. *Lapack Users' Guide.* SIAM, Philadelphia, 1992.

John M. Chambers. *Programming with Data.* Springer, New York, 1998. ISBN 0-387-98503-4.

John M. Chambers and Trevor J. Hastie. *Statistical Models in S.* Chapman & Hall, London, 1992. ISBN 0-412-83040-X.

Saikat DebRoy and Douglas M. Bates. Computational methods for multiple level linear mixed-effects models. Technical Report 1076, Department of Statistics, University of Wisconsin-Madison, 2003a.

Saikat DebRoy and Douglas M. Bates. Computational methods for single level linear mixed-effects models. Technical Report 1073, Department of Statistics, University of Wisconsin-Madison, 2003b.

José C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-PLUS.* Springer, 2000. ISBN 0-387-98957-9.

Stephen W. Raudenbush and Anthony S. Bryk. *Hierarchical Linear Models: Applications and Data Analysis Methods.* Sage, second edition, 2002. ISBN 0-7619-1904-X.

Germán Rodriguez and Noreen Goldman. An assessment of estimation procedures for multilevel models with binary responses. *Journal of the Royal Statistical Society, Series A, General*, 158:73–89, 1995.